Uncovering SAP vulnerabilities: Reversing and breaking the Diag protocol

Martin Gallo – Core Security
Defcon 20 – July 2012
Agenda

• Introduction
• Motivation and related work
• SAP Netweaver architecture and protocols layout
• Dissecting and understanding the Diag protocol
• Results and findings
• Defenses and countermeasures
• Conclusion and future work
Introduction
Introduction

• Leader business software provider
• Sensitive enterprise business processes runs on SAP systems
• SAP security became a hot topic
• Some components still not well covered
• Proprietary protocols used at different components
Introduction

- Dynamic Information and Action Gateway (Diag) protocol (aka “SAP GUI protocol”)
- Link between presentation layer (SAP GUI) and application layer (SAP Netweaver)
- Present in every SAP NW ABAP AS
- Compressed but unencrypted by default
- TCP ports 3200 to 3298
Agenda

- Introduction
- Motivation and related work
- SAP Netweaver architecture and protocols layout
- Dissecting and understanding the Diag protocol
- Results and findings
- Defenses and countermeasures
- Conclusion and future work
Motivation and related work
Previous work on Diag protocol

- Proprietary tools
- Sniffing through reflection-method
- Compression algorithm disclosed
- Proxy-like tool
- Decompression Wireshark plug-in
- Cain & Abel sniffing
Motivation

- Previous work mostly focused on decompression
- Protocol inner workings remains unknown
- No practical tool for penetration testing

Only 2 out of ~2300 security fixes published by SAP since 2009 affected components related to Diag
Agenda

- Introduction
- Motivation and related work
- SAP Netweaver architecture and protocols layout
- Dissecting and understanding the Diag protocol
- Results and findings
- Defenses and countermeasures
- Conclusion and future work
SAP Netweaver architecture and protocols layout
SAP Netweaver architecture
Relevant concepts and components

- **ABAP**
  - SAP’s programming language

- **Dispatcher and work processes (wp)**
  - **Dispatcher**: distribute user requests across wp
  - **Work processes**: handles specific tasks
    - Types: *dialog*, spool, update, background, lock

- **Dialog processing**
  - Programming method used by ABAP
  - Separates business programs in *screens* and *dialog steps*
SAP Protocols layout

Proprietary protocols

NI (Network Interface) Protocol

Standard protocols

HTTP
SSL

RFC
Diag Protocol
Router
BAPI
SOAP
Agenda

- Introduction
- Motivation and related work
- SAP Netweaver architecture and protocols layout
- Dissecting and understanding the Diag protocol
- Results and findings
- Defenses and countermeasures
- Conclusion and future work
Dissecting and understanding the Diag protocol
Dissecting and understanding the Diag protocol

Approach

• ‘Black-box’
• No binary reverse engineering techniques were used
• Enable system/developer traces (GUI/app server)
• Analyze network and application traces
• Learn by interacting with the components (GUI/app server)
• Continuous improvement of test tools based on gained knowledge
Dissecting and understanding the Diag protocol

NI (Network Interface) Protocol

Diag Protocol:
- DP Header (optional)
- Diag Header
- Compressio n Header (optional)
- Payload
  - Diag Item 1
  - ...
  - Diag Item n
Dissecting and understanding the Diag protocol

Initialization

- Identified only two relevant protocol states:
  - Not initialized
  - Initialized
    - User’s context assigned in shared memory
- Started by GUI application
- Only first packet
- Always uncompressed
Dissecting and understanding the Diag protocol

DP Header

- 200 bytes length
- Two different semantics
  - **IPC** (inter process communication)
    - Used in communications between dispatcher and work processes
    - Synchronization and status
  - **Network**
    - Most fields filled with default values
    - Relevant fields:
      - Terminal name, Length
- Only present during initialization (first packet)
Dissecting and understanding the Diag protocol

**Diag Header**

0 1 2 3 4 5 6 7

- **Mode**: Identifies different sessions using the same channel.
- **Comm Flag**: Compression enabled/disabled, encryption using SNC.
- **Mode Stat**: Diag Item 1...
- **Error Flag**: Diag Item n
- **Msg type**: Compression Header (optional)
- **Msg Info**: Diag Item 1
- **Msg RC**: payload
- **Comp Flag**: ...
Dissecting and understanding the Diag protocol

Compression

• Enabled by default
• Uses two variants of *Lempel-Ziv Adaptive Compression Algorithm*
  • *LZH* (Lempel-Ziv-Huffman) LZ77
  • *LZC* (Lempel-Ziv-Welch-Thomas) LZ78
• Same implementation as SAP’s *MaxDB* open source project
• Can be disabled in GUI by setting *TDW_NOCOMPRESS* environment variable
Dissecting and understanding the Diag protocol

Compression Header

- Uncompressed length
- Comp Alg
- Magic Bytes (x1F x9D)
- Special Byte

LZH: 0x12
LZC: 0x10
Dissecting and understanding the Diag protocol

Payload

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SES</td>
<td>Fixed length (16 bytes)</td>
<td>Session information</td>
</tr>
<tr>
<td>ICO</td>
<td>Fixed length (20 bytes)</td>
<td>Icon information</td>
</tr>
<tr>
<td>TIT</td>
<td>Fixed length (3 bytes)</td>
<td>Title information</td>
</tr>
<tr>
<td>DiagMessage</td>
<td>Fixed length (76 bytes)</td>
<td>Old Diag message</td>
</tr>
<tr>
<td>OKC</td>
<td>(?) Bytes</td>
<td></td>
</tr>
<tr>
<td>CHL</td>
<td>Fixed length (22 bytes)</td>
<td></td>
</tr>
<tr>
<td>SBA</td>
<td>Fixed length (9 bytes)</td>
<td>List items</td>
</tr>
<tr>
<td>EOM</td>
<td>Fixed length (0 bytes)</td>
<td>End of message</td>
</tr>
<tr>
<td>APPL/APPL4</td>
<td>Variable length</td>
<td></td>
</tr>
<tr>
<td>DIAG_XMLBlob</td>
<td>Variable length</td>
<td>XML Blob</td>
</tr>
<tr>
<td>SBA2</td>
<td>Fixed length (36 bytes)</td>
<td>List items</td>
</tr>
</tbody>
</table>
Dissecting and understanding the Diag protocol

APPL/APPL4 items

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>1 byte</td>
<td>APPL: 0x10, APPL4: 0x12</td>
</tr>
<tr>
<td>Length</td>
<td>3 bytes</td>
<td>APPL: 2 bytes, APPL4: 4 bytes</td>
</tr>
<tr>
<td>ID</td>
<td>1 byte</td>
<td></td>
</tr>
<tr>
<td>SID</td>
<td>2 bytes</td>
<td></td>
</tr>
</tbody>
</table>

NI (Network Interface) Protocol

- Diag Protocol
  - Payload
  - Diag Item 1
  - ...
Diag protocol security highlights

Protocol version
• APPL item included in payload during initialization
• Can disable compression using version number “200”

Authentication
• Performed as a regular dialog step
• Set user’s context on work processes shared memory

Embedded RFC calls
• APPL item that carries RFC calls in both directions
• Server doesn’t accept RFC calls until authenticated
Agenda

• Introduction
• Motivation and related work
• SAP Netweaver architecture and protocols layout
• Dissecting and understanding the Diag protocol
• Results and findings
• Defenses and countermeasures
• Conclusion and future work
Results and findings
Packet dissection

- Wireshark plug-in written in C/C++
  - NI Protocol dissector
    - TCP reassembling
  - Router Protocol dissector
    - Basic support
  - Diag protocol dissector
    - Decompression
    - DP header / Diag Header / Compression Header
    - Item ID/SID identification and dissection of relevant items
    - Call RFC dissector for embedded calls
  - RFC protocol dissector
    - Basic coverage of relevant parts
Packet dissection
Packet crafting

- Scapy classes
  - SAPNi
  - SAPDiagDP (DP Header)
  - SAPDiag (Diag header + compression)
  - SAPDiagItem
  - Custom classes for relevant Diag items

- PoC and example scripts
  - Information gathering
  - Login Brute Force
  - Proxy/MITM script
  - Diag server
Fuzzing approach

• Fuzzing scheme using
  • scapy classes
    • test cases generation
    • delivery
  • windbg
    • monitoring
  • xmlrpc
    • synchronization

• Monitoring of all work processes
Vulnerabilities found

• 6 vulnerabilities released on May 2012 affecting SAP NW 7.01/7.02, fix available on SAP Note 168710

• Unauthenticated remote denial of service when developed traces enabled
  • CVE-2012-2511 – DiagTraceAtoms function
  • CVE-2012-2512 – DiagTraceStreaml function
  • CVE-2012-2612 – DiagTraceHex function
Vulnerabilities found

• Unauthenticated remote denial of service
  • CVE-2012-2513 – Diaginput function
  • CVE-2012-2514 – DiagiEventSource function

• Unauthenticated remote code execution when developer traces enabled
  • CVE-2012-2611 – DiagTraceR3Info function
    • Stack-based buffer overflow while parsing ST_R3INFO CODEPAGE item
    • Thanks to Francisco Falcon (@fdfalcon) for the exploit
Attack scenarios

Target applications servers

- Exploit mentioned CVEs
- Gather server information
- Login brute force

Attacker

SAP NW AS
Attack scenarios

Target GUI users

- Rogue Server
- GUI User
- GUI Shortcut
- Attacker
- MitM
- Gather credentials
- Inject RFC calls in user’s GUI
- SAP NW AS
Agenda

• Introduction
• Motivation and related work
• SAP Netweaver architecture and protocols layout
• Dissecting and understanding the Diag protocol
• Results and findings
• Defenses and countermeasures
• Conclusion and future work
Defenses and countermeasures
Defenses and countermeasures

• Restrict network access to dispatcher service
  • TCP ports 3200-3298
  • Use application layer gateways

• Implement SNC client encryption
  • Provides authentication and encryption
  • Available for free at SAP Marketplace since 2011
  • See SAP Note 1643878

• Restrict use of GUI shortcuts
  • SAP GUI > 7.20 disabled by default
  • See SAP Note 1397000
Defenses and countermeasures

• Use WebGUI with HTTPS
  • See SAP Note 314568

• Patch regularly
  • Patch Tuesday
  • RSECNOTE program, see SAP Note 888889

• Patch CVEs affecting Diag
  • Look at CORE’s advisory for mitigation/countermeasures
  • See SAP Note 168710

• Test regularly
Agenda

- Introduction
- Motivation and related work
- SAP Netweaver architecture and protocols layout
- Dissecting and understanding the Diag protocol
- Results and findings
- Defenses and countermeasures
- Conclusion and future work
Conclusion and future work
Conclusion

- Protocol details now available to the security community
- Practical tools for dissection and crafting of protocol’s messages published
- New vectors for testing and assessing SAP environments
- Discussed countermeasures and defenses
Future work

• Security assessment and fuzzing of GUI/app server.
• Complete dissection of embedded RFC calls.
• Full implementation of attack scenarios
• Integration with external libraries and exploitation tools.
• Security assessment of SNC and coverage of encrypted traffic.
Thank you!

Thanks to Diego, Flavio, Dana, Wata and Euge
References

https://service.sap.com/sap/support/notes/1643879
http://www.secaron.de/Content/presse/fachartikel/sniffing_diag.pdf
http://conus.info/RE-articles/sapgui.html
http://www.sensepost.com/labs/conferences/2011/systems_application_proxy_pwnage
http://www.oxid.it/index.html
https://service.sap.com/securynotes
http://www.wireshark.org/
http://www.secdev.org/projects/scapy/
http://www.coresecurity.com/content/sap-netweaver-dispatcher-multiple-vulnerabilities
https://service.sap.com/sap/support/notes/1687910
http://help.sap.com/saphelp_nw70ehp2/helpdata/en/47/cc212b3fa5296fe10000000a42189b/frameset.htm
https://service.sap.com/sap/support/notes/1643878
https://service.sap.com/sap/support/notes/1397000
https://service.sap.com/sap/support/notes/314568
https://service.sap.com/sap/support/notes/888889