

Slide 1



WebSphere Education

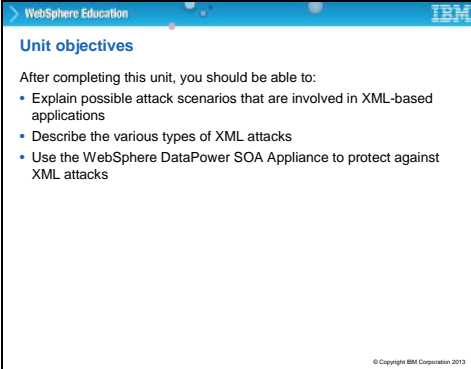
IBM

XML threat protection

© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

The slide features a blue header bar with the text 'WebSphere Education' on the left and the IBM logo on the right. The main content area is white with the title 'XML threat protection' centered. In the bottom left corner, there is a decorative graphic consisting of several overlapping circles in shades of blue and green, with small dots scattered around them. At the bottom center, there is a small copyright notice: '© Copyright IBM Corporation 2013' and 'Course materials may not be reproduced in whole or in part without the prior written permission of IBM.'

Slide 2



WebSphere Education

Unit objectives

After completing this unit, you should be able to:

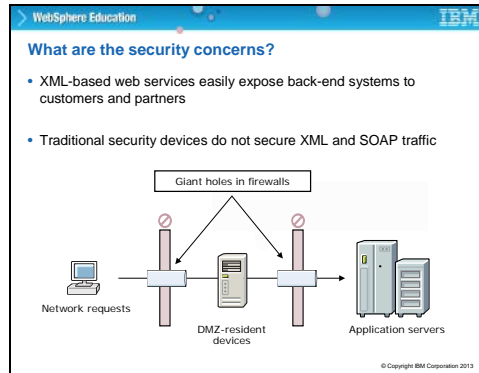
- Explain possible attack scenarios that are involved in XML-based applications
- Describe the various types of XML attacks
- Use the WebSphere DataPower SOA Appliance to protect against XML attacks

© Copyright IBM Corporation 2013

Unit overview

This unit teaches you how to defend against many possible attacks that are launched through XML.

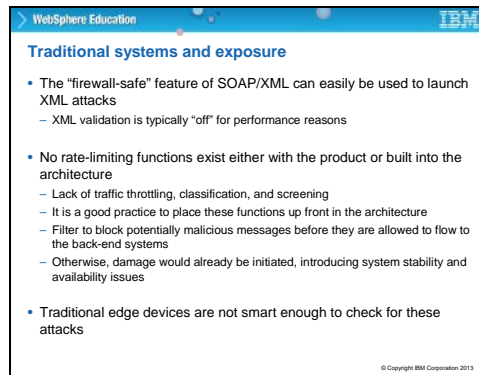
Slide 3



What are the security concerns?

So, what are these security concerns?

Most businesses nowadays want to expose their XML-based web services to customers and partners to maximize their revenue flow. This, in turn, requires that their firewalls have openings through which HTTP traffic can flow, which is an open invitation to hackers to exploit these holes. Since traditional security devices do not understand XML and SOAP traffic, it is relatively easy to send various threats and attacks.



WebSphere Education

Traditional systems and exposure

- The "firewall-safe" feature of SOAP/XML can easily be used to launch XML attacks
 - XML validation is typically "off" for performance reasons
- No rate-limiting functions exist either with the product or built into the architecture
 - Lack of traffic throttling, classification, and screening
 - It is a good practice to place these functions up front in the architecture
 - Filter to block potentially malicious messages before they are allowed to flow to the back-end systems
 - Otherwise, damage would already be initiated, introducing system stability and availability issues
- Traditional edge devices are not smart enough to check for these attacks

© Copyright IBM Corporation 2013

Traditional systems and exposure

The "firewall-safe" feature of SOAP/XML can easily be used to launch XML attacks since XML validation is typically "off" for performance reasons.

Also, most existing systems have no rate-limiting functions built into their architecture.

There is a lack of traffic throttling, classification, and screening.

The processing should occur up front in the architecture.

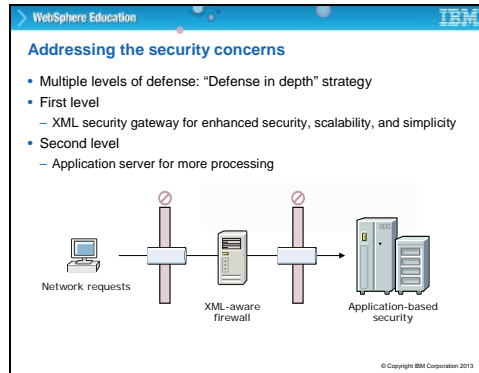
Potentially malicious messages should be filtered before they are allowed to flow to the back-end systems; otherwise damage would already be initiated, introducing system stability and availability issues.

Traditional edge devices are not smart enough to check for these attacks

Since these devices do not know about the data content, SOAP and XML can be transported to the back-end systems unhindered. And since XML validation is time-consuming, it is usually turned off in the back-end system to improve performance.

Some XML attacks rely upon sending messages at a certain rate, which is prevented by some rate-limiting function. But typical firewalls are not that sophisticated, so the potential exists to have such attacks reach the back-end systems.

Slide 5

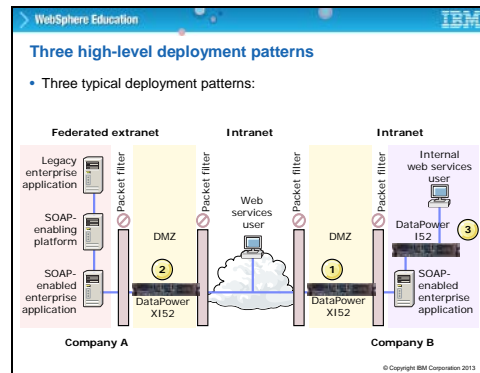


Addressing the security concerns

One way to address these concerns is to adopt a multi-layered approach. Let the firewalls continue to what they are good at, and add an extra device to do the more sophisticated checking of the message contents.

Use DataPower boxes at the "First level" in this diagram, since they can quickly and accurately detect and prevent content-based threats. At the "Second level", allow a back-end server to do some additional security processing. That second level can also be another DataPower system that plays a different role.

Slide 6



Three high-level deployment patterns

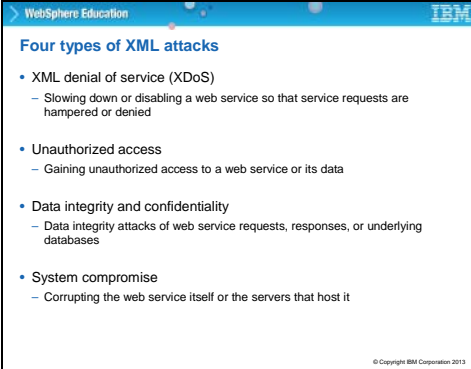
Here is an example of how to use DataPower boxes in a typical business setting. Number 1 is sitting inside the DMZ, assuming "Company A". It accepts all incoming data from the outside world, coming from the internet. It examines the content of all messages, looking for malicious data, and preventing attacks of various kinds. Now, this internet data can be coming from users or customers, sitting at home on a browser, connecting by using SSL over HTTPS or plain HTTP. DataPower supports encryption, authentication, and authorization, to ensure that users are valid and are doing legitimate work with.

Now, it can be that another company, say a business partner, is connecting over the internet. Perhaps they have their own DataPower box, number 2 in this diagram.

That provides a way to encrypt or sign the data that sent, and the DataPower box is configured to understand that and unscramble it. Perhaps their box inserts security tokens that are used for verification. There are a number of different scenarios that can be set up between two DataPower boxes in this way.

The third box in this diagram is handling connections that come from within your own company, providing authentication and authorization services. You might ask why you do that; surely you trust your own employees? Not so: Gartner did a survey several years ago that revealed that over 70% of all computer fraud is perpetrated by insiders. So you **do** need to protect your systems internally as well.

Slide 7



WebSphere Education

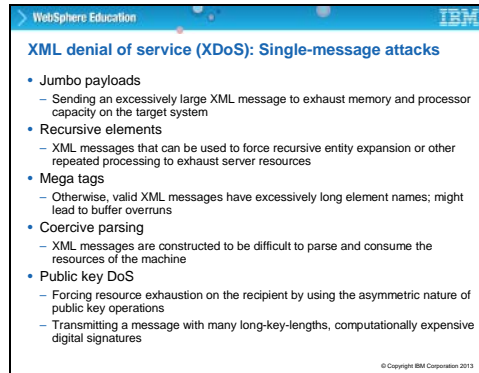
Four types of XML attacks

- XML denial of service (XDoS)
 - Slowing down or disabling a web service so that service requests are hampered or denied
- Unauthorized access
 - Gaining unauthorized access to a web service or its data
- Data integrity and confidentiality
 - Data integrity attacks of web service requests, responses, or underlying databases
- System compromise
 - Corrupting the web service itself or the servers that host it

© Copyright IBM Corporation 2013

Four types of XML attacks

Reviewing XML threats in more detail, they fall into these four broad categories. The first category is an XML denial of service, which is the slowing down or disabling of a web service so that service requests are hampered or denied. The second is unauthorized access a web service or its data. The third is, data integrity and confidentiality attacks can be made against web service requests, responses, or underlying databases. Fourthly, system compromise is the corrupting of the web service itself or the servers that host it.



WebSphere Education IBM

XML denial of service (XDoS): Single-message attacks

- Jumbo payloads
 - Sending an excessively large XML message to exhaust memory and processor capacity on the target system
- Recursive elements
 - XML messages that can be used to force recursive entity expansion or other repeated processing to exhaust server resources
- Mega tags
 - Otherwise, valid XML messages have excessively long element names; might lead to buffer overruns
- Coercive parsing
 - XML messages are constructed to be difficult to parse and consume the resources of the machine
- Public key DoS
 - Forcing resource exhaustion on the recipient by using the asymmetric nature of public key operations
 - Transmitting a message with many long-key-lengths, computationally expensive digital signatures

© Copyright IBM Corporation 2013

XML denial of service (XDoS): Single-message attacks

XML denial of service can take the form of single-message attacks or multiple message attacks. Look at single-message attacks first.

A jumbo payload is a large XML message that is intended to exhaust memory and CPU on the target system.

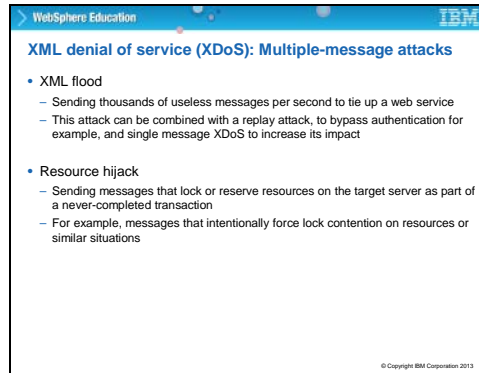
XML messages can contain recursive elements that force the parser to do recursive entity expansion or some other repeated processing to exhaust server resources.

There is a lab on recursive elements later, so you can see how that works.

Mega tags refer to XML messages that have valid syntax, but have with excessively long element names, which might lead to buffer overruns.

Coercive parsing uses techniques that are valid within the definition of XML, but are difficult to parse, such as recursive nesting, in which many elements, which all have the same name, are nested within each other. The intent is to construct valid XML messages that are difficult to parse to consume resources of the machine.

Regarding the Public Key DOS attack, you might remember how SSL works, with all those handshakes, by using asymmetric keys? Imagine if a hacker requests an SSL connection to your system by using a cipher suite that lists a single algorithm that is the most complex and requests a key that the longest. The extra time that is needed to handle that complexity might impact your systems performance. Now imagine that they drop the link when it is established, and try connecting again the same way. And they do repeat the process over and over. It is a good way to slow your system, yet it appears to be a valid process.



The slide is titled "XML denial of service (XDoS): Multiple-message attacks" and is part of a WebSphere Education presentation. It contains two main bullet points: "XML flood" and "Resource hijack". The "XML flood" bullet point includes two sub-points: "Sending thousands of useless messages per second to tie up a web service" and "This attack can be combined with a replay attack, to bypass authentication for example, and single message XDoS to increase its impact". The "Resource hijack" bullet point includes two sub-points: "Sending messages that lock or reserve resources on the target server as part of a never-completed transaction" and "For example, messages that intentionally force lock contention on resources or similar situations". The IBM logo is in the top right corner, and the copyright notice "© Copyright IBM Corporation 2013" is in the bottom right corner.

WebSphere Education

IBM

XML denial of service (XDoS): Multiple-message attacks

- XML flood
 - Sending thousands of useless messages per second to tie up a web service
 - This attack can be combined with a replay attack, to bypass authentication for example, and single message XDoS to increase its impact
- Resource hijack
 - Sending messages that lock or reserve resources on the target server as part of a never-completed transaction
 - For example, messages that intentionally force lock contention on resources or similar situations

© Copyright IBM Corporation 2013

XML denial of service (XDoS): Multiple-message attacks

An XML flood involves sending thousands of useless messages per second to tie up a web service. This attack can be combined with a replay attack to bypass authentication for example, and single message XML denial of service to increase its impact.

A resource hijack involves sending messages that lock or reserve resources on the target server as part of a never-completed transaction. For example, messages that intentionally force lock contention on resources or similar situations. The hijack can be initiated by someone with inside knowledge who knows how your database tables are interconnected, and knows how to call for certain pieces of data that causes the maximum table locking. Although, even an outsider can cause problems by, for example, filling an electronic shopping cart with hundreds or even thousands of items, but never checking out, making those items unavailable for sale.



WebSphere Education

Unauthorized access attacks

- Dictionary attacks
 - Guessing the password of a valid user by a brute force search through dictionary words
- Falsified messages
 - Faking that a message is from a valid user
 - Using an intermediary to gain a valid message and then modifying it to send a different message
- Replay attack
 - Resending a previously valid message for malicious effect
 - Possibly where only parts of the message (such as the security token) are replayed

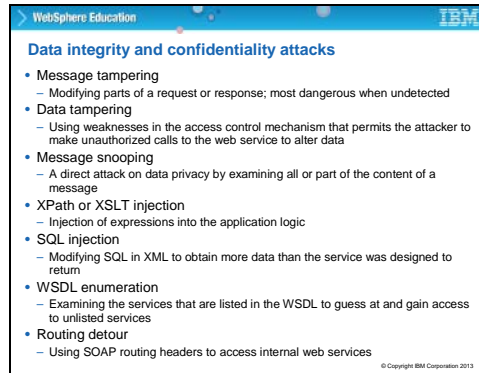
© Copyright IBM Corporation 2013

Unauthorized access attacks

Dictionary attacks involve guessing the password of a valid user by using a brute force search through dictionary words. The attack can easily be prevented by blocking the client after a certain number of attempts.

Falsified messages can be created by using a “man in the middle” attack to gain a valid message and then modifying it to send a different message.

A replay attack is resending a previously valid message for malicious effect, possibly where only parts of the message (such as the security token) are replayed.



The slide is titled "Data integrity and confidentiality attacks" and lists the following attacks:

- Message tampering
 - Modifying parts of a request or response; most dangerous when undetected
- Data tampering
 - Using weaknesses in the access control mechanism that permits the attacker to make unauthorized calls to the web service to alter data
- Message snooping
 - A direct attack on data privacy by examining all or part of the content of a message
- XPath or XSLT injection
 - Injection of expressions into the application logic
- SQL injection
 - Modifying SQL in XML to obtain more data than the service was designed to return
- WSDL enumeration
 - Examining the services that are listed in the WSDL to guess at and gain access to unlisted services
- Routing detour
 - Using SOAP routing headers to access internal web services

© Copyright IBM Corporation 2013

Data integrity and confidentiality attacks

Here are some typical data integrity and confidentiality attacks.

Message tampering is

Modifying parts of a request or response in flight, most dangerous when undetected

Data tampering is

Using weaknesses in the access control mechanism that permits the attacker to make unauthorized calls to the web service to alter data

Message snooping is

A direct attack on data privacy by examining all or part of the content of a message

XPath or XSLT injection is the

Injection of expressions into the application logic

SQL injection is

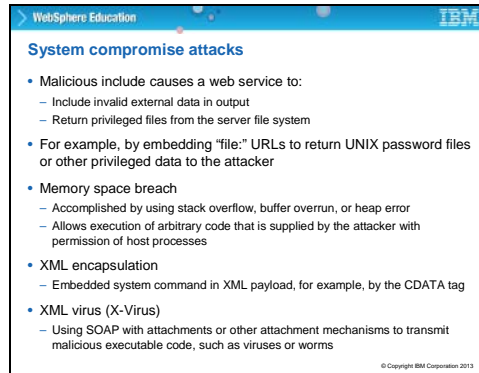
Modifying SQL in XML to obtain more data than the service was designed to return

WSDL enumeration is

Examining the services that are listed in the WSDL to guess at and gain access to unlisted services

And routing detour, which is

Using SOAP routing headers to access internal web services



The slide is titled 'System compromise attacks' and is part of a 'WebSphere Education' presentation. It lists several types of attacks:

- Malicious include causes a web service to:
 - Include invalid external data in output
 - Return privileged files from the server file system
- For example, by embedding "file:" URLs to return UNIX password files or other privileged data to the attacker
- Memory space breach
 - Accomplished by using stack overflow, buffer overrun, or heap error
 - Allows execution of arbitrary code that is supplied by the attacker with permission of host processes
- XML encapsulation
 - Embedded system command in XML payload, for example, by the CDATA tag
- XML virus (X-Virus)
 - Using SOAP with attachments or other attachment mechanisms to transmit malicious executable code, such as viruses or worms

© Copyright IBM Corporation 2013

System compromise attacks

A malicious include can cause a web service to Include invalid external data in output, which can then return privileged files from the server file system. For example, by using embedded "file:" URLs to return UNIX password files or other privileged data to the attacker.

Memory space breach is accomplished by using stack overflow, buffer overrun, or heap error, which allows execution of arbitrary code that is supplied by the attacker with permission of host processes. The process usually requires some specialized knowledge of the type of server that is used, and an understanding of how the operating system handles executable code.

XML encapsulation allows a hacker to embed system commands in an XML payload, for example, by using the CDATA tag.

XML virus is a way of using SOAP with attachments or other attachment mechanisms to transmit malicious programs, such as viruses or worms.

WebSphere Education

XML parser limits

- In the "Configure XML Manager" page, select the **XML Parser** tab
- This choice enhances security and stability by protecting against DoS attacks and runaway data
- XML parser limits:
 - Impose limits on various characteristics of XML documents that are parsed by the device.
- Parser limits are assigned to an XML manager, and any service that is supported by that manager inherits these limits
 - Note: Service-specific settings can override these inherited properties

XML Manager: default [us]

Apply Cancel Undo

These settings do not impact resource allocation, and are used only as part of your configuration.

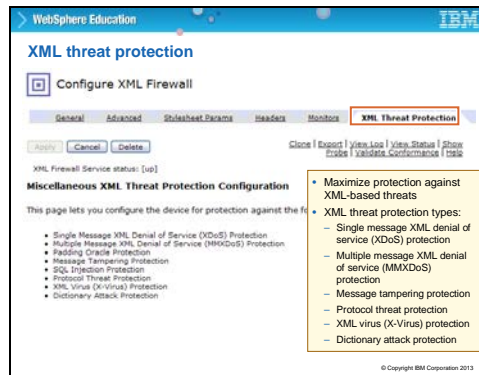
XML Bytes Scanned	4194304
XML Element Depth	512
XML Attribute Count	128
XML Maximum Node Size	33554432
XML Maximum Distinct Prefixes	0
XML Maximum Distinct Namespaces	0
XML Maximum Distinct Local Names	0
XML External Reference Handling	Forbidden

© Copyright IBM Corporation 2013

XML parser limits

The screen capture shows where you can set parameters to protect against some of these attacks. Services commonly use the default XML manager. The parameters can be overridden with a specific service.

In the Configure XML manager page, select the XML Parser tab. You can enhance security and stability by protecting against DoS attacks and runaway data. You can set XML parser limits on various characteristics of XML documents that are parsed by the device. These limits are assigned to an XML manager, and any service that is supported by that manager inherits these limits. Service-specific settings can override the inherited properties.



XML threat protection

The XML Threat Protection tab allows you to set parameters to guard against certain types of threat, such as:

- Single message XML denial of service (XDoS)
- Multiple message XML denial of service (MMXDoS)
- Message tampering
- Protocol threats
- XML virus (X-Virus)
- Dictionary attacks

WebSphere Education

XML threat protection: Single message XDoS

- XML threat protection settings:

Single Message XML Denial of Service (XDoS) Protection

Max. Message Size: KB

Override XML Manager parameter: ☐ on ☒ off

Max. XML Attribute Count: *

Max. XML Bytes Received: bytes *

Max. XML Element Depth: *

Max. XML Node Size: bytes *

XML Maximum Distinct Prefixes: *

XML Maximum Distinct Namespaces: *

XML Maximum Distinct Local Names: *

Attachment Byte Count Limit: bytes *

Attachment Package Byte Count Limit: bytes *

Recursive Entity Protection: ☐ on ☒ off

© Copyright IBM Corporation 2013

XML threat protection: Single message XDoS

The screen capture shows where you can set parameters that place limits on the data you handle. For example, you can specify maximum overall message size, maximum number of attributes within an element tag, maximum level of child processes within child processes, and many more.

The screenshot displays the 'XML threat protection: Multiple message XDoS' configuration page in the WebSphere Education console. The page includes a list of settings and a corresponding configuration form.

XML threat protection settings:

- Protects against a denial of service attack with multiple XML messages sent to a service
- XML threat protection settings:
 - Max. Duration for a Request
 - Interval for Measuring Request Rate from Host
 - Max. Request Rate from Host
 - Interval for Measuring Request Rate for Firewall
 - Maximum Request Rate for Firewall
 - Block Interval
 - Log Level

Multiple Message XML Denial of Service (MMXDoS) Protection

Enabling MMXDoS will create duration and count monitors and attach them to this firewall.

Enable MMXDoS Protection: ☒ on ☐ off

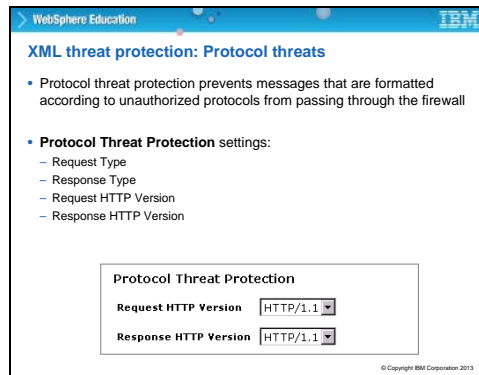
Max. Duration for a Request	<input type="text" value=""/>	max
Interval for Measuring Request Rate from Host	<input type="text" value="1000"/>	max
Max. Request Rate from Host	<input type="text" value=""/>	messages/interval
Interval for Measuring Request Rate for Firewall	<input type="text" value="1000"/>	max
Max. Request Rate for Firewall	<input type="text" value=""/>	messages/interval
Block Interval	<input type="text" value="0"/>	max
Log Level	<input type="text" value="OFF"/>	

© Copyright IBM Corporation 2013

XML threat protection: Multiple message XDoS

The Multiple Message XML Denial of Service section allows you to set various parameters, such as:

- Maximum Duration for a Request
- Interval for Measuring Request Rate from Host
- Maximum Request Rate from Host
- Interval for Measuring Request Rate for Firewall
- Maximum Request Rate for Firewall
- Block Interval
- Log Level



The slide is titled "XML threat protection: Protocol threats" and is part of a WebSphere Education presentation. It contains a bulleted list of settings for Protocol Threat Protection. Below the list is a screenshot of the "Protocol Threat Protection" configuration window, which shows two dropdown menus: "Request HTTP Version" and "Response HTTP Version", both set to "HTTP/1.1".

WebSphere Education

XML threat protection: Protocol threats

- Protocol threat protection prevents messages that are formatted according to unauthorized protocols from passing through the firewall
- **Protocol Threat Protection** settings:
 - Request Type
 - Response Type
 - Request HTTP Version
 - Response HTTP Version

Protocol Threat Protection

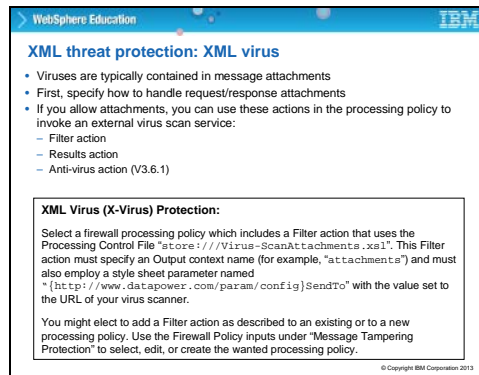
Request HTTP Version HTTP/1.1

Response HTTP Version HTTP/1.1

© Copyright IBM Corporation 2013

XML threat protection: Protocol threats

Sometimes, hackers pretend they are sending XML formatted to a particular version standard, but actually include elements that belong to another level. Protocol threat protection prevents messages that are formatted according to unauthorized protocols from passing through the firewall. Protocol Threat Protection settings include Request Type, Response Type, Request HTTP Version, and Response HTTP Version.



WebSphere Education

XML threat protection: XML virus

- Viruses are typically contained in message attachments
- First, specify how to handle request/response attachments
- If you allow attachments, you can use these actions in the processing policy to invoke an external virus scan service:
 - Filter action
 - Results action
 - Anti-virus action (V3.6.1)

XML Virus (X-Virus) Protection:

Select a firewall processing policy which includes a Filter action that uses the Processing Control File "store:///Virus-ScanAttachments.xml". This Filter action must specify an Output context name (for example, "attachments") and must also employ a style sheet parameter named "{http://www.datapower.com/param/config}SendTo" with the value set to the URL of your virus scanner.

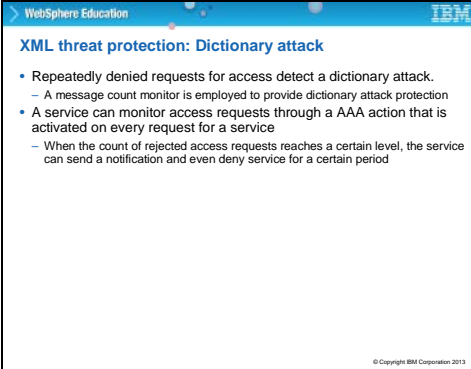
You might elect to add a Filter action as described to an existing or to a new processing policy. Use the Firewall Policy inputs under "Message Tampering Protection" to select, edit, or create the wanted processing policy.

© Copyright IBM Corporation 2013

XML threat protection: XML virus

Viruses are typically contained in message attachments, and when they are encoded as XML attachments, they are harder to detect because they are no longer in binary form. Although DataPower does not have any on-board virus-checking capability, it can take certain steps toward removing viruses. First, you can specify how to handle a request/response attachment. You can allow it to be processed as part of the message. You can reject outright any message that has an attachment. You can stream the attachment straight through to the back-end server and allow it work out what to do with it. You can strip off and discard the attachment, or you can allow the attachment to pass through unprocessed.

If you do allow attachments to be passed into the service, you can use one or more of the following in the processing policy to call an external virus-scanning service. Which is a Filter action, a Results action, or, you can call an Anti-virus action that passes the message to an external virus checking server.



WebSphere Education

XML threat protection: Dictionary attack

- Repeatedly denied requests for access detect a dictionary attack.
 - A message count monitor is employed to provide dictionary attack protection
- A service can monitor access requests through a AAA action that is activated on every request for a service
 - When the count of rejected access requests reaches a certain level, the service can send a notification and even deny service for a certain period

© Copyright IBM Corporation 2013

XML threat protection: Dictionary attack

Dictionary attacks are identified by repeatedly denied requests for access, which can be detected by using a message count monitor. You then configure an AAA action that is activated on every request for a service and supply the name of the message count monitor as the “Rejected Counter” within the AAA parameters. When the count of rejected access requests reaches a certain level, the service can send a notification and even deny service for a certain time.

WebSphere Education

Message tampering


- Message tampering protection can employ:
 - **Encrypt** and **decrypt** the message to hide the content
 - **Sign** and **verify** the message to ensure message integrity
 - **Validate** the schema of the message content for proper structure

The diagram illustrates a message flow process. At the top, a row of icons represents various actions: Filter, Sign, Verify, Validate, Encrypt, Decrypt, Transform, Route, AAA, Security, and Advanced. Below this, a horizontal flow line starts with a 'CLIENT' icon on the left and ends with a 'SERVER' icon on the right. The flow line includes several intermediate steps represented by icons: a diamond (likely a connector or gateway), a circle with a 'C' (likely a connector or gateway), a square with a 'V' (likely a connector or gateway), a square with a 'V' (likely a connector or gateway), and a square with a 'V' (likely a connector or gateway). The flow line is labeled 'CLIENT' at the start and 'SERVER' at the end. The diagram is credited to '© Copyright IBM Corporation 2013'.

Message tampering

Message tampering protection can employ:

- Encrypt and decrypt the message to hide the content
- Sign and verify the message to ensure message integrity
- Validate the schema of the message content for proper structure
- These functions can be programmed into a DataPower service by including the appropriate actions on the rule.

WebSphere Education


SQL injection attack

- A hacking technique that attempts to pass SQL commands through a web application or web service for execution by a back-end database
- Enabled by application code that does not properly screen SQL statement data that are received from a client or web form input (for example, SQL keywords, statements, comments)
- Example: database query from a web form
 - Expected: `SELECT id FROM logins WHERE userName='Tom'`
 - Actual: `SELECT id FROM logins WHERE userName='Tom' OR 'y'='y'`
- Example: building a database query from a web form
 - Expected: `SELECT product FROM products WHERE pName LIKE '%Chairs'`
 - Actual: `SELECT product FROM products WHERE pName LIKE '%Chairs' UNION SELECT username FROM dba_users WHERE username like '%'`

© Copyright IBM Corporation 2013


SQL injection attack

The SQL Injection Attack is a hacking technique that attempts to pass SQL commands through a web application or web service for execution by a back-end database. Application code enables it that does not properly screen SQL statement data that is received from a client or web form input (for example, SQL keywords, statements, comments).

For example, say that you have a database query from a web form that expects to create an SQL statement like `SELECT id FROM logins WHERE userName='Tom'`. In this example, the name "Tom" is entered by the user and transferred to the appropriate place in the SQL statement. But if the user adds more valid SQL syntax to the end of the name field, it can end up looking like `SELECT id FROM logins WHERE userName='Tom' OR 'y'='y'` which would return all the names on the database, not just Tom.


Another example might be a product catalog query that should look like `SELECT product FROM products WHERE pName LIKE '%Chairs'`. However, the hacker can add some extra syntax. The request can end up like `SELECT product FROM products WHERE pName LIKE '%Chairs' UNION SELECT username FROM dba_users WHERE username like '%'` which would return a list of valid database administrator user names. Since many back-end web server applications are written in haste, or written by improperly trained developers, the potential for SQL injection attacks is common. These attacks are on the increase. IBMs X-Force security team surveyed customers in May 2008, and counted about 2,500 SQL injection attacks per day. By midsummer 2009, that number was about 600,000 attacks per day on average.

- In July 2012 a hacker group reportedly stole 450,000 login credentials from Yahoo!. The logins were stored in plain text and were allegedly taken from a Yahoo subdomain, Yahoo! Voices. The group breached Yahoo's security by using a "union-based SQL injection technique".
- On October 1, 2012, a hacker group called "Team GhostShell" published the personal records of students, faculty, employees, and alumni from 53 universities that include Harvard, Princeton, Stanford, Cornell, Johns Hopkins, and the University of Zurich on pastebin.com. The hackers claimed that they were trying to "raise awareness towards the changes that are made in today education", bemoaning changing education laws in Europe and increases in tuition in the United States.

WebSphere Education 

SQL injection attack protection

- DataPower SQL injection attack protection uses a style-sheet-based approach to filter out potentially risky SQL strings
 - Add a **Filter** action to a firewall processing policy
 - Configure the processing control file to use `store:///SQL-Injection-Filter.xsl`
 - The style sheet sets a parameter named `{http://www.datapower.com/param/config}SQLPatternFile` to `store:///SQL-Injection-Patterns.xml`
 - You might customize it to point to a custom SQL injection pattern file

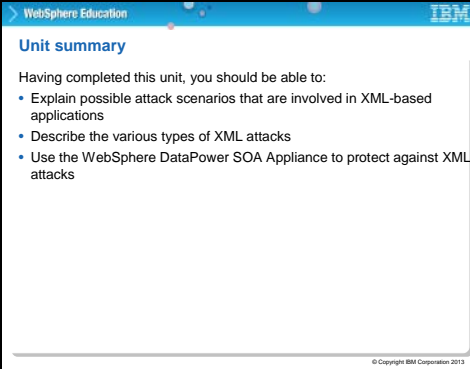


© Copyright IBM Corporation 2013

SQL injection attack protection

DataPower SQL injection attack protection uses a style sheet-based approach to filter out potentially risky SQL strings. All that you do is add a Filter action to a service processing policy and configure the filter to use SQL-Injection-Filter.xsl, which sets a parameter named SQLPatternFile to point to SQL-Injection-Patterns.xml. You might customize it to point to your own custom SQL injection pattern file.

Slide 23



WebSphere Education

IBM

Unit summary

Having completed this unit, you should be able to:

- Explain possible attack scenarios that are involved in XML-based applications
- Describe the various types of XML attacks
- Use the WebSphere DataPower SOA Appliance to protect against XML attacks

© Copyright IBM Corporation 2013

Slide 24

WebSphere Education

IBM

Checkpoint questions

1. What are the four types of XML threats?
 - A. XML denial of service (XDoS),
 - B. Unauthorized access
 - C. Data integrity and confidentiality
 - D. System compromise
 - E. Phishing
2. True or False: XML virus protection is a periodic virus scan utility that is offered by the DataPower SOA appliance.
3. True or False: The **Validate** action is sufficient to protect against a message tamper.

© Copyright IBM Corporation 2013

Slide 25

WebSphere Education


IBM

Checkpoint answers

1. A, B, C, and D. What are the four types of XML threats?
 - ✓ A. XML denial of service (XDoS),
 - ✓ B. Unauthorized access
 - ✓ C. Data integrity and confidentiality
 - ✓ D. System compromise
 - E. Phishing
2. False.
3. False. You also need the appropriate use of the **Sign** and **Verify** actions.

© Copyright IBM Corporation 2013

Slide 26



WebSphere Education

IBM

Exercise

Protecting against XML threats

© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 27

WebSphere Education

IBM

Exercise objectives

After completing this exercise, you should be able to:

- Run a recursive entity attack simulation
- Perform a recursive entity threat protection test
- Enable excessive attribute count threat protection
- Enable SQL injection attack prevention

© Copyright IBM Corporation 2013

Slide 28

