<pre><text>.GET_SIGNED8(< n></text></pre>
<pre>n> sequence of 8-bit signed integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised.</pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre>sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised.</pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre> <pre></pre>
<pre>(text>.GET_UNSIGNED 8(<n>)</n></pre> Treats the string of bytes represented by text as a sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised. Treats the text string returned by the prefix as a string of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The parameter makes allo area of the values of the values that the parameter of the value of n to 8. The parameter of the value of n to 8. The parameter of the value of no 8. The parameter of the values of the values of the values that the value of the values of the value of the values of
<pre>value outside of the turrent text, an UNDEFCONDITION IS raised. </pre> <pre>(text>.GET_UNSIGNED Sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised. </pre> <pre>(text>.GET_SIGNED16 Treats the text string returned by the prefix as a string (<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITILE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The parameter aligner aligner aligner aligner the value of n to 8. The parameter aligner aligner aligner the value of n to 8. The parameter aligner of a same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The parameter aligner aligner aligner aligner the value of n to 8. The parameter aligner aligner aligner aligner the you was the the other aligner aligner aligner the you was the the you was the parameter aligner the you was the parameter bloce the you was the parameter aligner aligner the you was aligner aligner aligner aligner aligner you was the you was the yout you the you was the you was the</endianness></n></pre>
<pre>(text>.GET_UNSIGNED 8(<n>) Treats the string of bytes represented by text as a sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised.</n></pre> (<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemotic value of LITILE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The parameter aligned to net on the value of n before the upgrade was 4, you must change the value of n to 8. The</endianness></n>
<pre><text>.GET_UNSIGNED Treats the string of bytes represented by text as a sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised. (<n>, <endianness>) Treats the text string returned by the prefix as a string of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The nameter and anners also en langent trices the values that</endianness></n></text></pre>
<pre>8(<n>) sequence of 8-bit unsigned integers and returns the integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised. </n></pre> <pre>(<n>, <endianness>) </endianness></n></pre> Treats the text string returned by the prefix as a string of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The nameter and innerse alor on botoms takes the walues that
<pre>integer at byte offset n. If the offset makes all or part of the value outside of the current text, an UNDEFcondition is raised. (<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITILE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The parameter and an ord or to 8. The parameter and an ord or an ord or to 8. The parameter and an ord ord no before the upgrade was 4, you must change the value of n to 8. The</endianness></n></pre>
<pre>of the value outside of the current text, an UNDEFcondition is raised. (<n>, <endianness>) (<n>, <endianness>) (<n>, <endianness>) Treats the text string returned by the prefix as a string of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITILE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The parameter and an outpart and the parameter by the same results and you and the parameter and you parameter and you parameter and you parameter by a here on the you before the upgrade</endianness></n></endianness></n></endianness></n></pre>
<pre>an UNDEFcondition is raised. <text>.GET_SIGNED16 (<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The parameter or offset results and you must that the same results and you must that the same results and you must results and you must that are results and you must r</endianness></n></text></pre>
<pre><text>.GET_SIGNED16 (<n>, <endianness>) Treats the text string returned by the prefix as a string of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITILE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The parameter andiances also ne largen takes the walkes that</endianness></n></text></pre>
<pre><text>.GET_SIGNED16 (<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n to 8. The napameter alignee to be the value of n to 8. The nap</endianness></n></text></pre>
<pre>(<n>, <endianness>) of bytes, extracts 16 bits starting at byte offset n, and converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The parameter endianness also no hereon takes the values that</endianness></n></pre>
converts the extracted bit sequence to a 16-bit signed integer. If the offset makes all or part of the value outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
outside of the current text, an UNDEF condition is raised. The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
The first parameter n is the byte offset from the current position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
position in the text string. Providing a byte offset enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
enables the function to handle items that are not aligned on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
on the boundaries that are required by indexes. The second parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
parameter, endianness, takes a mnemonic value of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2, the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
the parameter n was an index into an array of 16-bit items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
items. In NetScaler 9.3, the parameter is a byte offset. Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
Therefore, if you used this function in NetScaler 9.2, after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The nanameter andianness also no lengen takes the values that
after you upgrade to NetScaler 9.3, you must change n to 2*n to obtain the same results as you did earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
earlier. For example, if the value of n before the upgrade was 4, you must change the value of n to 8. The
was 4, you must change the value of n to 8. The
han and an and annass also no longon takes the values that
LUARAWELER POULADUESS ALSO DO LODPER LAKES LOP VALUES TOAT
it did in NetScaler 9.2, which were 0 and 1.
Instead, endianness accepts the mnemonic values mentioned
earlier. Example: HTTP.REQ.BODY(100).GET_SIGNED16(8,
BIG_ENDIAN).
<text>.GET_UNSIGNED Treats the text string returned by the prefix as a string</text>
16(<n>, of bytes, extracts 16 bits starting at byte offset n, and</n>
<pre><endianness>) converts the extracted bit sequence to a 16-bit unsigned</endianness></pre>
integer. If the offset makes all or part of the value
outside of the current text, an UNDEF condition is raised.
ine first parameter n is the byte offset from the current
position in the text string. Providing a Dyte offset
on the boundaries that are required by indexes. The second

	parameter, endianness, takes a mnemonic value
	of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2,
	the parameter n was an index into an array of 16-bit
	items. In NetScaler 9.3, the parameter is a byte offset.
	Therefore, if you used this function in NetScaler 9.2,
	after you upgrade to NetScaler 9.3, you must
	change n to 2*n to obtain the same results as you did
	earlier. For example, if the value of n before the upgrade
	was 4 you must change the value of $n \to 8$. The
	narameter endianness also no longer takes the values that
	it did in NotCoolon 0.2, which were 0 and 1
	The trade and improve accepts the memory is welled most improve
	Instead, endianness accepts the mnemonic values mentioned
	earlier. Example: HTTP.REQ.BODY(100).GET_UNSIGNED16(8,
	LITTLE_ENDIAN)
<text>.GET_SIGNED32</text>	Treats the text string returned by the prefix as a string
<pre>(<n>, <endianness>)</endianness></n></pre>	of bytes, extracts 32 bits starting at byte offset n, and
	converts the extracted bit sequence to a 32-bit signed
	integer. If the offset makes all or part of the value
	outside of the current text, an UNDEF condition is raised.
	The first parameter n is the byte offset from the current
	position in the text string Providing a byte offset
	enables the function to handle items that are not aligned
	on the houndaries that are neguined by indexes. The second
	on the boundaries that are required by indexes. The second
	parameter, englanness, takes a mnemonic value
	of LITTLE_ENDIAN or BIG_ENDIAN. Note: In NetScaler 9.2,
	the parameter n was an index into an array of 32-bit
	items. In NetScaler 9.3, the parameter is a byte offset.
	Therefore, if you used this function in NetScaler 9.2,
	after you upgrade to NetScaler 9.3, you must
	change n to 4*n to obtain the same results as you did
	earlier. For example, if the value of n before the upgrade
	was 4, you must change the value of n to 16. The
	narameter endianness also no longer takes the values that
	it did in NetScaler 9.2 which were 0 and 1
	Instead and appace accepts the memorie values mentioned
	anglier Examples UTD DEC DODY(1999) CET CICNED22(12)
	earlier. Example: HITP.REQ.BODY(1000).GET_SIGNED32(12,
	BIG_ENDIAN)
<text>.GET_UNSIGNED</text>	Treats the text string returned by the prefix as a string
32(<n>,</n>	of bytes, extracts 32 bits starting at byte offset n, and
<endianness>)</endianness>	returns the extracted bit sequence as part of a 64-bit
	unsigned long integer. If the offset makes all or part of
	the value outside of the current text, an UNDEFcondition
	is raised. The first parameter n is the byte offset from
	the current position in the text string. Providing a byte
	offset enables the function to handle items that are not

aligned on the boundaries that are required by indexes.
The second parameter, endianness, takes a mnemonic value
of LITTLE_ENDIAN or BIG_ENDIAN. Example: HTTP.REQ.BODY(100
0).GET UNSIGNED32(30, LITTLE ENDIAN