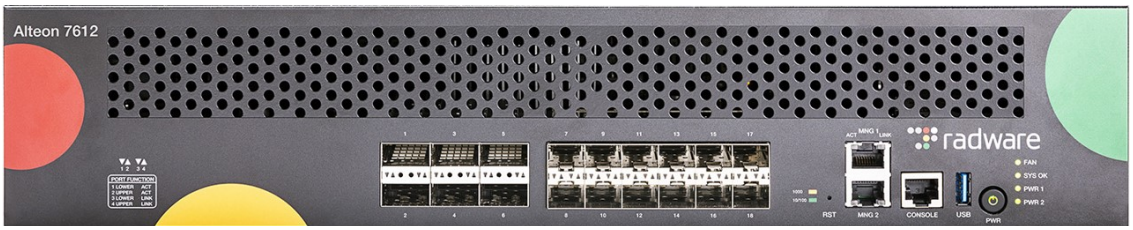


Alteon
APPSHAPE™++ REFERENCE GUIDE

Software Version 34.5.4, 34.0.9, 33.5.13, 33.0.17

Document ID: RDWR-ALOS-SDK-RG2506
June, 2025

Important Notices

The following important notices are presented in English, French, and German.

Important Notices

This guide is delivered subject to the following conditions and restrictions:

The AppShape++ Script Files provided by Radware Ltd. are subject to the Special License Terms included in each of the electronic AppShape++ Script Files and are also subject to Radware's End User License Agreement, a copy of which (as may be amended from time to time) can be found at the end of this document or at http://www.radware.com/Resources/eula.html.

Please note that if you create your own scripts using any AppShape++ Scripts provided by Radware, such self-created scripts are not controlled by Radware and therefore Radware will not be liable for any malfunctions resulting from such self-created scripts.

The copyright and all other intellectual property rights and trade secrets included in this guide are owned by Radware Ltd.

The guide is provided to Radware customers for the sole purpose of obtaining information with respect to the installation and use of the Radware products described in this document, and may not be used for any other purpose.

The information contained in this guide is proprietary to Radware and must be kept in strict confidence.

It is strictly forbidden to copy, duplicate, reproduce or disclose this guide or any part thereof without the prior written consent of Radware.

Notice importante

Ce guide est sujet aux conditions et restrictions suivantes:

Les applications AppShape++ Script Files fournies par Radware Ltd. sont soumises aux termes de la Licence Spéciale ("Special License Terms") incluse dans chaque fichier électronique "AppShape++ Script Files" mais aussi au Contrat de Licence d'Utilisateur Final de Radware qui peut être modifié de temps en temps et dont une copie est disponible à la fin du présent document ou à l'adresse suivante: http://www.radware.com/Resources/eula.html.

Nous attirons votre attention sur le fait que si vous créez vos propres fichiers de commande (fichiers "script") en utilisant l'application "AppShape++ Script Files" fournie par Radware, ces fichiers "script" ne sont pas contrôlés par Radware et Radware ne pourra en aucun cas être tenue responsable des dysfonctionnements résultant des fichiers "script" ainsi créés.

Le copyright ainsi que tout autre droit lié à la propriété intellectuelle et aux secrets industriels contenus dans ce guide sont la propriété de Radware Ltd.

Ce guide d'informations est fourni à nos clients dans le cadre de l'installation et de l'usage des produits de Radware décrits dans ce document et ne pourra être utilisé dans un but autre que celui pour lequel il a été conçu.

Les informations répertoriées dans ce document restent la propriété de Radware et doivent être conservées de manière confidentielle.

Il est strictement interdit de copier, reproduire ou divulguer des informations contenues dans ce manuel sans avoir obtenu le consentement préalable écrit de Radware.

Wichtige Anmerkung

Dieses Handbuch wird vorbehaltlich folgender Bedingungen und Einschränkungen ausgeliefert:

Die von Radware Ltd bereitgestellten AppShape++ Scriptdateien unterliegen den in jeder elektronischen AppShape++ Scriptdatei enthalten besonderen Lizenzbedingungen sowie Radware's Endbenutzer-Lizenzvertrag (von welchem eine Kopie in der jeweils geltenden Fassung am Ende dieses Dokuments oder unter http://www.radware.com/Resources/eula.html erhältlich ist).

Bitte beachten Sie, dass wenn Sie Ihre eigenen Skripte mit Hilfe eines von Radware bereitgestellten AppShape++ Skripts erstellen, diese selbsterstellten Skripte nicht von Radware kontrolliert werden und Radware daher keine Haftung für Funktionsfehler übernimmt, welche von diesen selbsterstellten Skripten verursacht werden.

Dieses Handbuch wird Kunden von Radware mit dem ausschließlichen Zweck ausgehändigt, Informationen zu Montage und Benutzung der in diesem Dokument beschriebene Produkte von Radware bereitzustellen. Es darf für keinen anderen Zweck verwendet werden.

Die in diesem Handbuch enthaltenen Informationen sind Eigentum von Radware und müssen streng vertraulich behandelt werden.

Es ist streng verboten, dieses Handbuch oder Teile daraus ohne vorherige schriftliche Zustimmung von Radware zu kopieren, vervielfältigen, reproduzieren oder offen zu legen.

Copyright Notices

The following copyright notices are presented in English, French, and German.

Copyright Notices

The programs included in this product are subject to a restricted use license and can only be used in conjunction with this application.

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of the OpenSSL License and the original SSLeay license apply to the toolkit. See below for the actual license texts. Actually both licenses are BSD-style Open Source licenses. In case of any license issues related to OpenSSL, please contact openssl-core@openssl.org.

OpenSSL License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)
4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.

5. Products derived from this software may not be called “OpenSSL” nor may “OpenSSL” appear in their names without prior written permission of the OpenSSL Project.
6. Redistributions of any form whatsoever must retain the following acknowledgment:

   “This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/)”

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT “AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Original SSLeay License

Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)

All rights reserved.

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com).

The implementation was written so as to conform with Netscapes SSL.

This library is free for commercial and non-commercial use as long as the following conditions are aheared to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed.

If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used.

This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

   "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the rouines from the library being used are not cryptographic related :-).
4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgment:

   "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS"' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publically available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

This product contains the Rijndael cipher

The Rijndael implementation by Vincent Rijmen, Antoon Bosselaers and Paulo Barreto is in the public domain and distributed with the following license:

@version 3.0 (December 2000)

Optimized ANSI C code for the Rijndael cipher (now AES)

@author Vincent Rijmen <vincent.rijmen@esat.kuleuven.ac.be>

@author Antoon Bosselaers <antoon.bosselaers@esat.kuleuven.ac.be>

@author Paulo Barreto <paulo.barreto@terra.com.br>

The OnDemand Switch may use software components licensed under the GNU General Public License Agreement Version 2 (GPL v.2) including LinuxBios and Filo open source projects. The source code of the LinuxBios and Filo is available from Radware upon request. A copy of the license can be viewed at: http://www.gnu.org/licenses/old-licenses/gpl-2.0.html.

This code is hereby placed in the public domain.

This product contains code developed by the OpenBSD Project

Copyright ©1983, 1990, 1992, 1993, 1995

The Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This product includes software developed by Markus Friedl.

This product includes software developed by Theo de Raadt.

This product includes software developed by Niels Provos

This product includes software developed by Dug Song

This product includes software developed by Aaron Campbell

This product includes software developed by Damien Miller

This product includes software developed by Kevin Steves

This product includes software developed by Daniel Kouril

This product includes software developed by Wesley Griffin

This product includes software developed by Per Allansson

This product includes software developed by Nils Nordman

This product includes software developed by Simon Wilkinson

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

This product contains work derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm. RSA Data Security, Inc. makes no representations concerning either the merchantability of the MD5 Message - Digest Algorithm or the suitability of the MD5 Message - Digest Algorithm for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

HiddenComment: The following text, which is yet to be translated into French and German, April 2015, from Hanna Mordichai, was added for APpppM. Vision only [Output attr=Vision_OLH Vision_UG].

This product includes the DB2 Express-C database, the copyrights of which are owned IBM.

HiddenComment: The following text is for AppWall [Output attr=appwall], which is yet to be translated into French and German. 26 January 2012 from Hanna Mordichai and Gadi M.

This product contains a Access Protocol API developed by The OpenLDAP Foundation titled "OpenLDAP Lightweight Directory Access Protocol API". Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved.

OpenLDAP is a registered trademark of the OpenLDAP Foundation.

The following license terms apply to the openLDAP Access Protocol API, including, without limitations, the below list of conditions and disclaimer:

The OpenLDAP Public License

Version 2.8, 17 August 2003

Redistribution and use of this software and associated documentation ("Software"), with or without modification, are permitted provided that the following conditions are met:

1. Redistributions in source form must retain copyright statements and notices,
2. Redistributions in binary form must reproduce applicable copyright statements and notices, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution, and
3. Redistributions must contain a verbatim copy of this document.

The OpenLDAP Foundation may revise this license from time to time. Each revision is distinguished by a version number. You may use this Software under terms of this license revision or under the terms of any subsequent revision of the license.

THIS SOFTWARE IS PROVIDED BY THE OPENLDAP FOUNDATION AND ITS CONTRIBUTORS "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OPENLDAP FOUNDATION, ITS CONTRIBUTORS, OR THE AUTHOR(S) OR OWNER(S) OF THE SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The names of the authors and copyright holders must not be used in advertising or otherwise to promote the sale, use or other dealing in this Software without specific, written prior permission. Title to copyright in this Software shall at all times remain with copyright holders.

OpenLDAP is a registered trademark of the OpenLDAP Foundation. Copyright 1999-2003 The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved. Permission to copy and distribute verbatim copies of this document is granted.

This product contains the ACE RADIUS library developed by Mr. Alex Agranov. Copyright ©2004-2009, Alex Agranov <alexagr@users.sourceforge.net> All rights reserved.

The ACE RADIUS library is licensed under BSD License, which allows its use both in open-source and commercial projects.

The license terms are as follows:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions, and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of ace-radius nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This software includes the SNMP++v3.2.25 library Copyright ©2001-2010 Jochen Katz, Frank Fock

The SNMP++v3.2.25 library is based on SNMP++2.6 from Hewlett Packard: Copyright ©1996 Hewlett-Packard Company

ATTENTION: USE OF THIS SOFTWARE IS SUBJECT TO THE FOLLOWING TERMS.

Permission to use, copy, modify, distribute and/or sell this software and/or its documentation is hereby granted without fee. User agrees to display the above copyright notice and this license notice in all copies of the software and any documentation of the software. User agrees to assume all liability for the use of the software; Hewlett-Packard and Jochen Katz make no representations about the suitability of this software for any purpose. It is provided “AS-IS” without warranty of any kind, either express or implied. User hereby grants a royalty-free license to any and all derivatives based upon this software code base.

Stuttgart, Germany, Thu Sep 2 00:07:47 CEST 2010

Notice traitant du copyright

Les programmes intégrés dans ce produit sont soumis à une licence d’utilisation limitée et ne peuvent être utilisés qu’en lien avec cette application.

Ce produit renferme des codes développés dans le cadre du projet OpenSSL.

Ce produit inclut un logiciel développé dans le cadre du projet OpenSSL. Pour un usage dans la boîte à outils OpenSSL (http://www.openssl.org/).

Copyright ©1998-2005 Le projet OpenSSL. Tous droits réservés. Ce produit inclut la catégorie de chiffre Rijndael.

L’implémentation de Rijindael par Vincent Rijmen, Antoon Bosselaers et Paulo Barreto est du domaine public et distribuée sous les termes de la licence suivante:

@version 3.0 (Décembre 2000)

Code ANSI C code pour Rijndael (actuellement AES)

@author Vincent Rijmen <vincent.rijmen@esat.kuleuven.ac.be>

@author Antoon Bosselaers <antoon.bosselaers@esat.kuleuven.ac.be>

@author Paulo Barreto <paulo.barreto@terra.com.br>.

Le commutateur OnDemand peut utiliser les composants logiciels sous licence, en vertu des termes de la licence GNU General Public License Agreement Version 2 (GPL v.2), y compris les projets à source ouverte LinuxBios et Filo. Le code source de LinuxBios et Filo est disponible sur demande auprès de Radware. Une copie de la licence est répertoriée sur: http://www.gnu.org/licenses/old-licenses/gpl-2.0.html.

Ce code est également placé dans le domaine public.

Ce produit renferme des codes développés dans le cadre du projet OpenSSL.

Copyright ©1983, 1990, 1992, 1993, 1995

Les membres du conseil de l'Université de Californie. Tous droits réservés.

La distribution et l'usage sous une forme source et binaire, avec ou sans modifications, est autorisée pour autant que les conditions suivantes soient remplies:

1. La distribution d'un code source doit inclure la notice de copyright mentionnée ci-dessus, cette liste de conditions et l'avis de non-responsabilité suivant.
2. La distribution, sous une forme binaire, doit reproduire dans la documentation et/ou dans tout autre matériel fourni la notice de copyright mentionnée ci-dessus, cette liste de conditions et l'avis de non-responsabilité suivant.
3. Le nom de l'université, ainsi que le nom des contributeurs ne seront en aucun cas utilisés pour approuver ou promouvoir un produit dérivé de ce programme sans l'obtention préalable d'une autorisation écrite.

Ce produit inclut un logiciel développé par Markus Friedl.

Ce produit inclut un logiciel développé par Theo de Raadt.

Ce produit inclut un logiciel développé par Niels Provos.

Ce produit inclut un logiciel développé par Dug Song.

Ce produit inclut un logiciel développé par Aaron Campbell.

Ce produit inclut un logiciel développé par Damien Miller.

Ce produit inclut un logiciel développé par Kevin Steves.

Ce produit inclut un logiciel développé par Daniel Kouril.

Ce produit inclut un logiciel développé par Wesley Griffin.

Ce produit inclut un logiciel développé par Per Allansson.

Ce produit inclut un logiciel développé par Nils Nordman.

Ce produit inclut un logiciel développé par Simon Wilkinson.

La distribution et l'usage sous une forme source et binaire, avec ou sans modifications, est autorisée pour autant que les conditions suivantes soient remplies:

1. La distribution d'un code source doit inclure la notice de copyright mentionnée ci-dessus, cette liste de conditions et l'avis de non-responsabilité suivant.
2. La distribution, sous une forme binaire, doit reproduire dans la documentation et/ou dans tout autre matériel fourni la notice de copyright mentionnée ci-dessus, cette liste de conditions et l'avis de non-responsabilité suivant.

LE LOGICIEL MENTIONNÉ CI-DESSUS EST FOURNI TEL QUEL PAR LE DÉVELOPPEUR ET TOUTE GARANTIE, EXPLICITE OU IMPLICITE, Y COMPRIS, MAIS SANS S'Y LIMITER, TOUTE GARANTIE IMPLICITE DE QUALITÉ MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER EST EXCLUE.

EN AUCUN CAS L'AUTEUR NE POURRA ÊTRE TENU RESPONSABLE DES DOMMAGES DIRECTS, INDIRECTS, ACCESSOIRES, SPÉCIAUX, EXEMPLAIRES OU CONSÉCUTIFS (Y COMPRIS, MAIS SANS S'Y LIMITER, L'ACQUISITION DE BIENS OU DE SERVICES DE REMPLACEMENT, LA PERTE D'USAGE, DE DONNÉES OU DE PROFITS OU L'INTERRUPTION DES AFFAIRES), QUELLE QU'EN SOIT LA CAUSE ET LA THÉORIE DE RESPONSABILITÉ, QU'IL S'AGISSE D'UN CONTRAT, DE RESPONSABILITÉ STRICTE OU D'UN ACTE DOMMAGEABLE (Y COMPRIS LA NÉGLIGENCE OU AUTRE), DÉCOULANT DE QUELLE QUE FAÇON QUE CE SOIT DE L'USAGE DE CE LOGICIEL, MÊME S'IL A ÉTÉ AVERTI DE LA POSSIBILITÉ D'UN TEL DOMMAGE.

Copyrightvermerke

Die in diesem Produkt enthalten Programme unterliegen einer eingeschränkten Nutzungslizenz und können nur in Verbindung mit dieser Anwendung benutzt werden.

Dieses Produkt enthält einen vom OpenSSL-Projekt entwickelten Code

Dieses Produkt enthält vom OpenSSL-Projekt entwickelte Software. Zur Verwendung im OpenSSL Toolkit (http://www.openssl.org/).

Copyright ©1998-2005 The OpenSSL Project. Alle Rechte vorbehalten. Dieses Produkt enthält die Rijndael cipher.

Die Rijndael-Implementierung von Vincent Rijndael, Anton Bosselaers und Paulo Barreto ist öffentlich zugänglich und wird unter folgender Lizenz vertrieben:

@version 3.0 (December 2000)

Optimierter ANSI C Code für den Rijndael cipher (jetzt AES)

@author Vincent Rijmen <vincent.rijmen@esat.kuleuven.ac.be>

@author Antoon Bosselaers <antoon.bosselaers@esat.kuleuven.ac.be>

@author Paulo Barreto <paulo.barreto@terra.com.br>

Der OnDemand Switch verwendet möglicherweise Software, die im Rahmen der DNU Allgemeine Öffentliche Lizenzvereinbarung Version 2 (GPL v.2) lizensiert sind, einschließlich LinuxBios und Filo Open Source-Projekte. Der Quellcode von LinuxBios und Filo ist bei Radware auf Anfrage erhältlich. Eine Kopie dieser Lizenz kann eingesehen werden unter http://www.gnu.org/licenses/old-licenses/gpl-2.0.html.

Dieser Code wird hiermit allgemein zugänglich gemacht.

Dieses Produkt enthält einen vom OpenBSD-Projekt entwickelten Code

Copyright ©1983, 1990, 1992, 1993, 1995

The Regents of the University of California. Alle Rechte vorbehalten.

Die Verbreitung und Verwendung in Quell- und binärem Format, mit oder ohne Veränderungen, sind unter folgenden Bedingungen erlaubt:

1. Die Verbreitung von Quellcodes muss den voranstehenden Copyrightvermerk, diese Liste von Bedingungen und den folgenden Haftungsausschluss beibehalten.
2. Die Verbreitung in binärem Format muss den voranstehenden Copyrightvermerk, diese Liste von Bedingungen und den folgenden Haftungsausschluss in der Dokumentation und/oder andere Materialien, die mit verteilt werden, reproduzieren.
3. Weder der Name der Universität noch die Namen der Beitragenden dürfen ohne ausdrückliche vorherige schriftliche Genehmigung verwendet werden, um von dieser Software abgeleitete Produkte zu empfehlen oder zu bewerben.

Dieses Produkt enthält von Markus Friedl entwickelte Software.

Dieses Produkt enthält von Theo de Raadt entwickelte Software.

Dieses Produkt enthält von Niels Provos entwickelte Software.

Dieses Produkt enthält von Dug Song entwickelte Software.

Dieses Produkt enthält von Aaron Campbell entwickelte Software.

Dieses Produkt enthält von Damien Miller entwickelte Software.

Dieses Produkt enthält von Kevin Steves entwickelte Software.

Dieses Produkt enthält von Daniel Kouril entwickelte Software.

Dieses Produkt enthält von Wesley Griffin entwickelte Software.

Dieses Produkt enthält von Per Allansson entwickelte Software.

Dieses Produkt enthält von Nils Nordman entwickelte Software.

Dieses Produkt enthält von Simon Wilkinson entwickelte Software.

Die Verbreitung und Verwendung in Quell- und binärem Format, mit oder ohne Veränderungen, sind unter folgenden Bedingungen erlaubt:

1. Die Verbreitung von Quellcodes muss den voranstehenden Copyrightvermerk, diese Liste von Bedingungen und den folgenden Haftungsausschluss beibehalten.
2. Die Verbreitung in binärem Format muss den voranstehenden Copyrightvermerk, diese Liste von Bedingungen und den folgenden Haftungsausschluss in der Dokumentation und/oder andere Materialien, die mit verteilt werden, reproduzieren.

SÄMTLICHE VORGENANNTE SOFTWARE WIRD VOM AUTOR IM IST-ZUSTAND (“AS IS”) BEREITGESTELLT. JEGLICHE AUSDRÜCKLICHEN ODER IMPLIZITEN GARANTIEN, EINSCHLIESSLICH, DOCH NICHT BESCHRÄNKT AUF DIE IMPLIZIERTEN GARANTIEN DER MARKTGÄNGIGKEIT UND DER ANWENDBARKEIT FÜR EINEN BESTIMMTEN ZWECK, SIND AUSGESCHLOSSEN.

UNTER KEINEN UMSTÄNDEN HAFTET DER AUTOR FÜR DIREKTE ODER INDIREKTE SCHÄDEN, FÜR BEI VERTRAGSERFÜLLUNG ENTSTANDENE SCHÄDEN, FÜR BESONDERE SCHÄDEN, FÜR SCHADENSERSATZ MIT STRAFCHARAKTER, ODER FÜR FOLGESCHÄDEN EINSCHLIESSLICH, DOCH NICHT BESCHRÄNKT AUF, ERWERB VON ERSATZGÜTERN ODER ERSATZLEISTUNGEN; VERLUST AN NUTZUNG, DATEN ODER GEWINN; ODER GESCHÄFTSUNTERBRECHUNGEN) GLEICH, WIE SIE ENTSTANDEN SIND, UND FÜR JEGLICHE ART VON HAFTUNG, SEI ES VERTRÄGE, GEFÄHRDUNGSHAFTUNG, ODER DELIKTISCHE HAFTUNG (EINSCHLIESSLICH FAHRLÄSSIGKEIT ODER ANDERE), DIE IN JEGLICHER FORM FOLGE DER BENUTZUNG DIESER SOFTWARE IST, SELBST WENN AUF DIE MÖGLICHKEIT EINES SOLCHEN SCHADENS HINGEWIESEN WURDE.

Standard Warranty

The following standard warranty is presented in English, French, and German.

Standard Warranty

Radware offers a limited warranty for all its products (“Products”). Radware hardware products are warranted against defects in material and workmanship for a period of one year from date of shipment. Radware software carries a standard warranty that provides bug fixes for up to 90 days after date of purchase. Should a Product unit fail anytime during the said period(s), Radware will, at its discretion, repair or replace the Product.

For hardware warranty service or repair, the product must be returned to a service facility designated by Radware. Customer shall pay the shipping charges to Radware and Radware shall pay the shipping charges in returning the product to the customer. Please see specific details outlined in the Standard Warranty section of the customer’s purchase order.

Radware shall be released from all obligations under its Standard Warranty in the event that the Product and/or the defective component has been subjected to misuse, neglect, accident or improper installation, or if repairs or modifications were made by persons other than Radware authorized service personnel, unless such repairs by others were made with the written consent of Radware.

The service provided by Radware is warranted solely pursuant to the provisions of the Service Level Agreement between Radware and you.

The service provided by Radware is warranted solely pursuant to the provisions of the Service Level Agreement between Radware and you.

The service provided by Radware is warranted solely pursuant to the provisions of the Service Level Agreement between Radware and you.

EXCEPT AS SET FORTH ABOVE, ALL RADWARE PRODUCTS (HARDWARE AND SOFTWARE) ARE PROVIDED BY “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

EXCEPT AS SET FORTH ABOVE, ALL RADWARE PRODUCTS AND SERVICES ARE PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

EXCEPT AS SET FORTH ABOVE, ALL RADWARE PRODUCTS AND SERVICES ARE PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

EXCEPT AS SET FORTH ABOVE, ALL RADWARE PRODUCTS AND SERVICES ARE PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED.

Garantie standard

Radware octroie une garantie limitée pour l'ensemble de ses produits ("Produits"). Le matériel informatique (hardware) Radware est garanti contre tout défaut matériel et de fabrication pendant une durée d'un an à compter de la date d'expédition. Les logiciels (software) Radware sont fournis avec une garantie standard consistant en la fourniture de correctifs des dysfonctionnements du logiciels (bugs) pendant une durée maximum de 90 jours à compter de la date d'achat. Dans l'hypothèse où un Produit présenterait un défaut pendant ladite (lesdites) période(s), Radware procédera, à sa discrétion, à la réparation ou à l'échange du Produit.

S'agissant de la garantie d'échange ou de réparation du matériel informatique, le Produit doit être retourné chez un réparateur désigné par Radware. Le Client aura à sa charge les frais d'envoi du Produit à Radware et Radware supportera les frais de retour du Produit au client. Veuillez consulter les conditions spécifiques décrites dans la partie "Garantie Standard" du bon de commande client.

Radware est libérée de toutes obligations liées à la Garantie Standard dans l'hypothèse où le Produit et/ou le composant défectueux a fait l'objet d'un mauvais usage, d'une négligence, d'un accident ou d'une installation non conforme, ou si les réparations ou les modifications qu'il a subi ont été effectuées par d'autres personnes que le personnel de maintenance autorisé par Radware, sauf si Radware a donné son consentement écrit à ce que de telles réparations soient effectuées par ces personnes.

Les seules garanties pour le service fourni par Radware sont celles stipulées dans le Contrat de Niveau de Service conclu entre Radware et vous.

Les seules garanties pour le service fourni par Radware sont celles stipulées dans le Contrat de Niveau de Service conclu entre Radware et vous.

Les seules garanties pour le service fourni par Radware sont celles stipulées dans le Contrat de Niveau de Service conclu entre Radware et vous.

SAUF DANS LES CAS PREVUS CI-DESSUS, L'ENSEMBLE DES PRODUITS RADWARE (MATERIELS ET LOGICIELS) SONT FOURNIS "TELS QUELS" ET TOUTES GARANTIES EXPRESSES OU IMPLICITES SONT EXCLUES, EN CE COMPRIS, MAIS SANS S'Y RESTREINDRE, LES GARANTIES IMPLICITES DE QUALITE MARCHANDE ET D'ADÉQUATION À UNE UTILISATION PARTICULIÈRE.

SAUF STIPULATION CONTRAIRE SUSMENTIONNÉE, TOUS LES PRODUITS ET SERVICES RADWARE SONT FOURNIS "EN L'ÉTAT" ET TOUTE GARANTIE, QU'ELLE SOIT EXPRESSE OU IMPLICITE, EN CE COMPRIS NOTAMMENT, LA GARANTIE IMPLICITE DE VALEUR MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER, EST EXCLUE.

SAUF STIPULATION CONTRAIRE SUSMENTIONNÉE, TOUS LES PRODUITS ET SERVICES RADWARE SONT FOURNIS "EN L'ÉTAT" ET TOUTE GARANTIE, QU'ELLE SOIT EXPRESSE OU IMPLICITE, EN CE COMPRIS NOTAMMENT, LA GARANTIE IMPLICITE DE VALEUR MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER, EST EXCLUE.

SAUF STIPULATION CONTRAIRE SUSMENTIONNÉE, TOUS LES PRODUITS ET SERVICES RADWARE SONT FOURNIS "EN L'ÉTAT" ET TOUTE GARANTIE, QU'ELLE SOIT EXPRESSE OU IMPLICITE, EN CE COMPRIS NOTAMMENT, LA GARANTIE IMPLICITE DE VALEUR MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER, EST EXCLUE.

Standard Garantie

Radware bietet eine begrenzte Garantie für alle seine Produkte ("Produkte") an. Hardware Produkte von Radware haben eine Garantie gegen Material- und Verarbeitungsfehler für einen Zeitraum von einem Jahr ab Lieferdatum. Radware Software verfügt über eine Standard Garantie zur Fehlerbereinigung für einen Zeitraum von bis zu 90 Tagen nach Erwerbsdatum. Sollte ein Produkt innerhalb des angegebenen Garantiezeitraumes einen Defekt aufweisen, wird Radware das Produkt nach eigenem Ermessen entweder reparieren oder ersetzen.

Für den Hardware Garantieservice oder die Reparatur ist das Produkt an eine von Radware bezeichnete Serviceeinrichtung zurückzugeben. Der Kunde hat die Versandkosten für den Transport des Produktes zu Radware zu tragen, Radware übernimmt die Kosten der Rückversendung des Produktes an den Kunden. Genauere Angaben entnehmen Sie bitte dem Abschnitt zur Standard Garantie im Bestellformular für Kunden.

Radware ist von sämtlichen Verpflichtungen unter seiner Standard Garantie befreit, sofern das Produkt oder der fehlerhafte Teil zweckentfremdet genutzt, in der Pflege vernachlässigt, einem Unfall ausgesetzt oder unsachgemäß installiert wurde oder sofern Reparaturen oder Modifikationen von anderen Personen als durch Radware autorisierten Kundendienstmitarbeitern vorgenommen wurden, es sei denn, diese Reparatur durch besagte andere Personen wurden mit schriftlicher Genehmigung seitens Radware durchgeführt.

Für die von Radware zu erbringende Leistung wird lediglich eine Gewährleistung anhand der Bestimmungen im Dienstleistungsvertrag, der zwischen Radware und Ihnen abgeschlossen wurde, übernommen.

Für die von Radware zu erbringende Leistung wird lediglich eine Gewährleistung anhand der Bestimmungen im Dienstleistungsvertrag, der zwischen Radware und Ihnen abgeschlossen wurde, übernommen.

Für die von Radware zu erbringende Leistung wird lediglich eine Gewährleistung anhand der Bestimmungen im Dienstleistungsvertrag, der zwischen Radware und Ihnen abgeschlossen wurde, übernommen.

MIT AUSNAHME DES OBEN DARGESTELLTEN, SIND ALLE RADWARE PRODUKTE (HARDWARE UND SOFTWARE) GELIEFERT "WIE GESEHEN" UND JEGLICHE AUSDRÜCKLICHEN ODER STILLSCHWEIGENDEN GARANTIEN, EINSCHLIESSLICH ABER NICHT BEGRENZT AUF STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTFÄHIGKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN ZWECK AUSGESCHLOSSEN.

SOFERN WEITER OBEN NICHTS ANDERES VEREINBART WURDE, WERDEN DIE VON RADWARE ZU LIEFERNDEN PRODUKTE UND ZU ERBRINGENDEN LEISTUNGEN "WIE BESEHEN" GELIEFERT/ ERBRACHT. AUSDRÜCKLICHE ODER STILLSCHWEIGENDE GEWÄHRLEISTUNGEN, EINSCHLIESSLICH, JEDOCH NICHT NUR, DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN SPEZIELLEN ZWECK, SIND AUSGESCHLOSSEN.

SOFERN WEITER OBEN NICHTS ANDERES VEREINBART WURDE, WERDEN DIE VON RADWARE ZU LIEFERNDEN PRODUKTE UND ZU ERBRINGENDEN LEISTUNGEN "WIE BESEHEN" GELIEFERT/ ERBRACHT. AUSDRÜCKLICHE ODER STILLSCHWEIGENDE GEWÄHRLEISTUNGEN, EINSCHLIESSLICH, JEDOCH NICHT NUR, DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN SPEZIELLEN ZWECK, SIND AUSGESCHLOSSEN.

SOFERN WEITER OBEN NICHTS ANDERES VEREINBART WURDE, WERDEN DIE VON RADWARE ZU LIEFERNDEN PRODUKTE UND ZU ERBRINGENDEN LEISTUNGEN "WIE BESEHEN" GELIEFERT/ ERBRACHT. AUSDRÜCKLICHE ODER STILLSCHWEIGENDE GEWÄHRLEISTUNGEN, EINSCHLIESSLICH, JEDOCH NICHT NUR, DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN SPEZIELLEN ZWECK, SIND AUSGESCHLOSSEN.

Limitations on Warranty and Liability

The following limitations on warranty and liability are presented in English, French, and German.

Limitations on Warranty and Liability

IN NO EVENT SHALL RADWARE LTD. OR ANY OF ITS AFFILIATED ENTITIES BE LIABLE FOR ANY DAMAGES INCURRED BY THE USE OF THE PRODUCTS (INCLUDING BOTH HARDWARE AND SOFTWARE) DESCRIBED IN THIS USER GUIDE, OR BY ANY DEFECT OR INACCURACY IN THIS USER GUIDE ITSELF. THIS INCLUDES BUT IS NOT LIMITED TO ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION). THE ABOVE LIMITATIONS WILL APPLY EVEN IF RADWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

Limitations de la Garantie et Responsabilité

RADWARE LTD. OU SES ENTITIES AFFILIES NE POURRONT EN AUCUN CAS ETRE TENUES RESPONSABLES DES DOMMAGES SUBIS DU FAIT DE L'UTILISATION DES PRODUITS (EN CE COMPRIS LES MATERIELS ET LES LOGICIELS) DECRITS DANS CE MANUEL D'UTILISATION, OU DU FAIT DE DEFAUT OU D'IMPRECISIONS DANS CE MANUEL D'UTILISATION, EN CE COMPRIS, SANS TOUTEFOIS QUE CETTE ENUMERATION SOIT CONSIDEREE COMME LIMITATIVE, TOUS DOMMAGES DIRECTS, INDIRECTS, ACCIDENTELS, SPECIAUX, EXEMPLAIRES, OU ACCESSOIRES (INCLUANT, MAIS SANS S'Y RESTREINDRE, LA FOURNITURE DE PRODUITS OU DE SERVICES DE REMPLACEMENT; LA PERTE D'UTILISATION, DE DONNEES OU DE PROFITS; OU L'INTERRUPTION DES AFFAIRES). LES LIMITATIONS CI-DESSUS S'APPLIQUERONT QUAND BIEN MEME RADWARE A ETE INFORMEE DE LA POSSIBLE EXISTENCE DE CES DOMMAGES. CERTAINES JURIDICTIONS N'ADMETTANT PAS LES EXCLUSIONS OU LIMITATIONS DE GARANTIES IMPLICITES OU DE RESPONSABILITE EN CAS DE DOMMAGES ACCESSOIRES OU INDIRECTS, LESDITES LIMITATIONS OU EXCLUSIONS POURRAIENT NE PAS ETRE APPLICABLE DANS VOTRE CAS.

Haftungs- und Gewährleistungsausschluss

IN KEINEM FALL IST RADWARE LTD. ODER EIN IHR VERBUNDENES UNTERNEHMEN HAFTBAR FÜR SCHÄDEN, WELCHE BEIM GEBRAUCH DES PRODUKTES (HARDWARE UND SOFTWARE) WIE IM BENUTZERHANDBUCH BESCHRIEBEN, ODER AUFGRUND EINES FEHLERS ODER EINER UNGENAUIGKEIT IN DIESEM BENUTZERHANDBUCH SELBST ENTSTANDEN SIND. DAZU GEHÖREN UNTER ANDEREM (OHNE DARAUF BEGRENZT ZU SEIN) JEGLICHE DIREKTEN; IDIREKTEN; NEBEN; SPEZIELLEN, BELEGTEN ODER FOLGESCHÄDEN (EINSCHLIESSLICH ABER NICHT BEGRENZT AUF BESCHAFFUNG ODER ERSATZ VON WAREN ODER DIENSTEN, NUTZUNGSAUSFALL, DATEN- ODER GEWINNVERLUST ODER BETRIEBSUNTERBRECHUNGEN). DIE OBEN GENANNTEN BEGRENZUNGEN GREIFEN AUCH, SOFERN RADWARE AUF DIE MÖGLICHKEIT EINES SOLCHEN SCHADENS HINGEWIESEN WORDEN SEIN SOLLTE. EINIGE RECHTSORDNUNGEN LASSEN EINEN AUSSCHLUSS ODER EINE BEGRENZUNG STILLSCHWEIGENDER GARANTIEN ODER HAFTUNGEN BEZÜGLICH NEBEN- ODER FOLGESCHÄDEN NICHT ZU, SO DASS DIE OBEN DARGESTELLTE BEGRENZUNG ODER DER AUSSCHLUSS SIE UNTER UMSTÄNDEN NICHT BETREFFEN WIRD.

Safety Instructions

The following safety instructions are presented in English, French, and German.

Safety Instructions

CAUTION

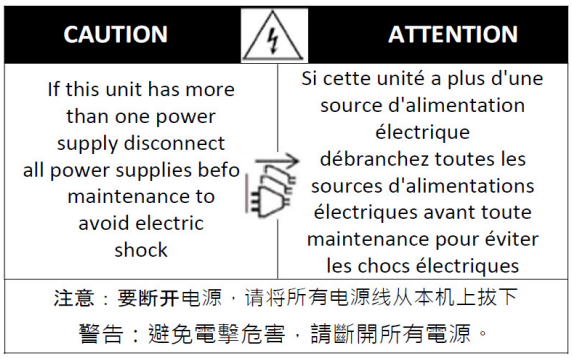
A readily accessible disconnect device shall be incorporated in the building installation wiring.

Due to the risks of electrical shock, and energy, mechanical, and fire hazards, any procedures that involve opening panels or changing components must be performed by qualified service personnel only.

To reduce the risk of fire and electrical shock, disconnect the device from the power line before removing cover or panels.

The following figure shows the caution label that is attached to Radware platforms with dual power supplies.

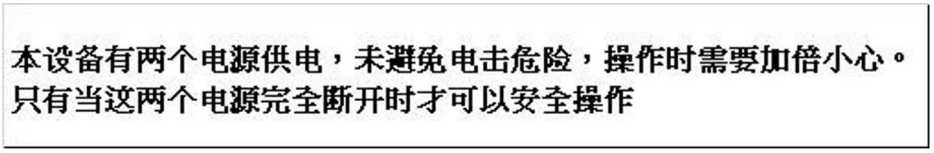
Figure 1: Electrical Shock Hazard Label



DUAL-POWER-SUPPLY-SYSTEM SAFETY WARNING IN CHINESE

The following figure is the warning for Radware platforms with dual power supplies.

Figure 2: Dual-Power-Supply-System Safety Warning in Chinese



本設備有兩個電源供電，為避免電擊危險，操作時需要加倍小心。
只有當這兩個電源完全斷開時才可以安全操作。”

Translation of Dual-Power-Supply-System Safety Warning in Chinese:

This unit has more than one power supply. Disconnect all power supplies before maintenance to avoid electric shock.

SERVICING

Do not perform any servicing other than that contained in the operating instructions unless you are qualified to do so. There are no serviceable parts inside the unit.

HIGH VOLTAGE

Any adjustment, maintenance, and repair of the opened instrument under voltage must be avoided as much as possible and, when inevitable, must be carried out only by a skilled person who is aware of the hazard involved.

Capacitors inside the instrument may still be charged even if the instrument has been disconnected from its source of supply.

GROUNDING

Before connecting this device to the power line, the protective earth terminal screws of this device must be connected to the protective earth in the building installation.

LASER

This equipment is a Class 1 Laser Product in accordance with IEC60825 - 1: 1993 + A1:1997 + A2:2001 Standard.

FUSES

Make sure that only fuses with the required rated current and of the specified type are used for replacement. The use of repaired fuses and the short-circuiting of fuse holders must be avoided. Whenever it is likely that the protection offered by fuses has been impaired, the instrument must be made inoperative and be secured against any unintended operation.

LINE VOLTAGE

Before connecting this instrument to the power line, make sure the voltage of the power source matches the requirements of the instrument. Refer to the Specifications for information about the correct power rating for the device.

48V DC-powered platforms have an input tolerance of 36-72V DC.

SPECIFICATION CHANGES

Specifications are subject to change without notice.

Note: This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15B of the FCC Rules and EN55022 Class A, EN 55024; EN 61000-3-2; EN 61000-3-3; IEC 61000 4-2 to 4-6, IEC 61000 4-8 and IEC 61000-4-11For CE MARK Compliance. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user is required to correct the interference at his own expense.

SPECIAL NOTICE FOR NORTH AMERICAN USERS

For North American power connection, select a power supply cord that is UL Listed and CSA Certified 3 - conductor, [18 AWG], terminated in a molded on plug cap rated 125 V, [10 A changedMar12], with a minimum length of 1.5m [six feet] but no longer than 4.5m...For European connection, select a power supply cord that is internationally harmonized and marked "<HAR>", 3 - conductor, 0,75 mm2 minimum mm2 wire, rated 300 V, with a PVC insulated jacket. The cord must have a molded on plug cap rated 250 V, 3 A.

RESTRICT AREA ACCESS

The DC powered equipment should only be installed in a Restricted Access Area.

INSTALLATION CODES

This device must be installed according to country national electrical codes. For North America, equipment must be installed in accordance with the US National Electrical Code, Articles 110 - 16, 110 -17, and 110 -18 and the Canadian Electrical Code, Section 12.

INTERCONNECTION OF UNITS

Cables for connecting to the unit RS232 and Ethernet Interfaces must be UL certified type DP-1 or DP-2. (Note- when residing in non LPS circuit)

OVERCURRENT PROTECTION

A readily accessible listed branch-circuit over current protective device rated 15 A must be incorporated in the building wiring for each power input.

REPLACEABLE BATTERIES

If equipment is provided with a replaceable battery, and is replaced by an incorrect battery type, then an explosion may occur. This is the case for some Lithium batteries and the following is applicable:

- If the battery is placed in an **Operator Access Area**, there is a marking close to the battery or a statement in both the operating and service instructions.
- If the battery is placed elsewhere in the equipment, there is a marking close to the battery or a statement in the service instructions.

This marking or statement includes the following text warning:

CAUTION

RISK OF EXPLOSION IF BATTERY IS REPLACED BY AN INCORRECT BATTERY TYPE. DISPOSE OF USED BATTERIES ACCORDING TO THE INSTRUCTIONS.

Caution – To Reduce the Risk of Electrical Shock and Fire

1. This equipment is designed to permit connection between the earthed conductor of the DC supply circuit and the earthing conductor equipment. See Installation Instructions.
2. All servicing must be undertaken only by qualified service personnel. There are not user serviceable parts inside the unit.
3. DO NOT plug in, turn on or attempt to operate an obviously damaged unit.
4. Ensure that the chassis ventilation openings in the unit are NOT BLOCKED.
5. Replace a blown fuse ONLY with the same type and rating as is marked on the safety label adjacent to the power inlet, housing the fuse.
6. Do not operate the device in a location where the maximum ambient temperature exceeds 40°C/104°F.
7. Be sure to unplug the power supply cord from the wall socket BEFORE attempting to remove and/or check the main power fuse.
   CLASS 1 LASER PRODUCT AND REFERENCE TO THE MOST RECENT LASER STANDARDS IEC 60 825-1:1993 + A1:1997 + A2:2001 AND EN 60825-1:1994+A1:1996+ A2:2001

AC units for Denmark, Finland, Norway, Sweden (marked on product):

- Denmark - “Unit is class I - unit to be used with an AC cord set suitable with Denmark deviations. The cord includes an earthing conductor. The Unit is to be plugged into a wall socket outlet which is connected to a protective earth. Socket outlets which are not connected to earth are not to be used!”
- Finland - (Marking label and in manual) - “Laite on liitettävä suojamaadoituskoskettimilla varustettuun pistorasiaan”
- Norway (Marking label and in manual) - “Apparatet må tilkoples jordet stikkontakt”
- Unit is intended for connection to IT power systems for Norway only.
- Sweden (Marking label and in manual) - “Apparaten skall anslutas till jordat uttag.”

To connect the power connection:

1. Connect the power cable to the main socket, located on the rear panel of the device.
2. Connect the power cable to the grounded AC outlet.

CAUTION

Risk of electric shock and energy hazard. Disconnecting one power supply disconnects only one power supply module. To isolate the unit completely, disconnect all power supplies.

Instructions de sécurité

AVERTISSEMENT

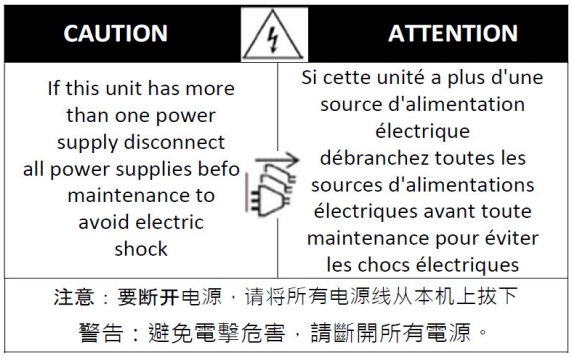
Un dispositif de déconnexion facilement accessible sera incorporé au câblage du bâtiment.

En raison des risques de chocs électriques et des dangers énergétiques, mécaniques et d'incendie, chaque procédure impliquant l'ouverture des panneaux ou le remplacement de composants sera exécutée par du personnel qualifié.

Pour réduire les risques d'incendie et de chocs électriques, déconnectez le dispositif du bloc d'alimentation avant de retirer le couvercle ou les panneaux.

La figure suivante montre l'étiquette d'avertissement apposée sur les plateformes Radware dotées de plus d'une source d'alimentation électrique.

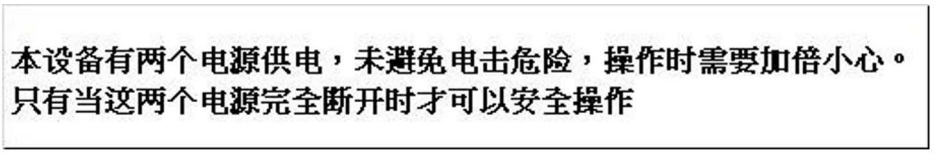
Figure 3: Étiquette d'avertissement de danger de chocs électriques



AVERTISSEMENT DE SÉCURITÉ POUR LES SYSTÈMES DOTÉS DE DEUX SOURCES D'ALIMENTATION ÉLECTRIQUE (EN CHINOIS)

La figure suivante représente l'étiquette d'avertissement pour les plateformes Radware dotées de deux sources d'alimentation électrique.

Figure 4: Avertissement de sécurité pour les systèmes dotes de deux sources d'alimentation électrique (en chinois)



本設備有兩個電源供電，為避免電擊危險，操作時需要加倍小心。
只有當這兩個電源完全斷開時才可以安全操作。”

Traduction de la Avertissement de sécurité pour les systèmes dotes de deux sources d'alimentation électrique (en chinois):

Cette unité est dotée de plus d'une source d'alimentation électrique. Déconnectez toutes les sources d'alimentation électrique avant d'entretenir l'appareil ceci pour éviter tout choc électrique.

ENTRETIEN

N'effectuez aucun entretien autre que ceux répertoriés dans le manuel d'instructions, à moins d'être qualifié en la matière. Aucune pièce à l'intérieur de l'unité ne peut être remplacée ou réparée.

HAUTE TENSION

Tout réglage, opération d'entretien et réparation de l'instrument ouvert sous tension doit être évité. Si cela s'avère indispensable, confiez cette opération à une personne qualifiée et consciente des dangers impliqués.

Les condensateurs au sein de l'unité risquent d'être chargés même si l'unité a été déconnectée de la source d'alimentation électrique.

MISE A LA TERRE

Avant de connecter ce dispositif à la ligne électrique, les vis de protection de la borne de terre de cette unité doivent être reliées au système de mise à la terre du bâtiment.

LASER

Cet équipement est un produit laser de classe 1, conforme à la norme IEC60825 - 1: 1993 + A1: 1997 + A2: 2001.

FUSIBLES

Assurez-vous que, seuls les fusibles à courant nominal requis et de type spécifié sont utilisés en remplacement. L'usage de fusibles réparés et le court-circuitage des porte-fusibles doivent être évités. Lorsqu'il est pratiquement certain que la protection offerte par les fusibles a été détériorée, l'instrument doit être désactivé et sécurisé contre toute opération involontaire.

TENSION DE LIGNE

Avant de connecter cet instrument à la ligne électrique, vérifiez que la tension de la source d'alimentation correspond aux exigences de l'instrument. Consultez les spécifications propres à l'alimentation nominale correcte du dispositif.

Les plateformes alimentées en 48 CC ont une tolérance d'entrée comprise entre 36 et 72 V CC.
MODIFICATIONS DES SPÉCIFICATIONS

Les spécifications sont sujettes à changement sans notice préalable.

Remarque: Cet équipement a été testé et déclaré conforme aux limites définies pour un appareil numérique de classe A, conformément au paragraphe 15B de la réglementation FCC et EN55022 Classe A, EN 55024, EN 61000-3-2; EN 61000-3-3; IEC 61000 4-2 to 4-6, IEC 61000 4-8, et IEC 61000-4-11, pour la marque de conformité de la CE. Ces limites sont fixées pour fournir une protection raisonnable contre les interférences nuisibles, lorsque l'équipement est utilisé dans un environnement commercial. Cet équipement génère, utilise et peut émettre des fréquences radio et, s'il n'est pas installé et utilisé conformément au manuel d'instructions, peut entraîner des interférences nuisibles aux communications radio. Le fonctionnement de cet équipement dans une zone résidentielle est susceptible de provoquer des interférences nuisibles, auquel cas l'utilisateur devra corriger le problème à ses propres frais.

NOTICE SPÉCIALE POUR LES UTILISATEURS NORD-AMÉRICAINS

Pour un raccordement électrique en Amérique du Nord, sélectionnez un cordon d'alimentation homologué UL et certifié CSA 3 - conducteur, [18 AWG], muni d'une prise moulée à son extrémité, de 125 V, [10 A changedMar12], d'une longueur minimale de 1,5 m [six pieds] et maximale de 4,5m...Pour la connexion européenne, choisissez un cordon d'alimentation mondialement homologué et marqué "<HAR>", 3 - conducteur, câble de 0,75 mm2 minimum, de 300 V, avec une gaine en PVC isolée. La prise à l'extrémité du cordon, sera dotée d'un sceau moulé indiquant: 250 V, 3 A.

ZONE A ACCÈS RESTREINT

L'équipement alimenté en CC ne pourra être installé que dans une zone à accès restreint.

CODES D'INSTALLATION

Ce dispositif doit être installé en conformité avec les codes électriques nationaux. En Amérique du Nord, l'équipement sera installé en conformité avec le code électrique national américain, articles 110-16, 110 -17, et 110 -18 et le code électrique canadien, Section 12.

INTERCONNEXION DES UNÎTES

Les câbles de connexion à l'unité RS232 et aux interfaces Ethernet seront certifiés UL, type DP-1 ou DP-2. (Remarque- s'ils ne résident pas dans un circuit LPS).

PROTECTION CONTRE LES SURCHARGES

Un circuit de dérivation, facilement accessible, sur le dispositif de protection du courant de 15 A doit être intégré au câblage du bâtiment pour chaque puissance consommée.

BATTERIES REMPLAÇABLES

Si l'équipement est fourni avec une batterie, et qu'elle est remplacée par un type de batterie incorrect, elle est susceptible d'exploser. C'est le cas pour certaines batteries au lithium, les éléments suivants sont donc applicables:

- Si la batterie est placée dans une zone d'accès opérateur, une marque est indiquée sur la batterie ou une remarque est insérée, aussi bien dans les instructions d'exploitation que d'entretien.
- Si la batterie est placée ailleurs dans l'équipement, une marque est indiquée sur la batterie ou une remarque est insérée dans les instructions d'entretien.

Cette marque ou remarque inclut l'avertissement textuel suivant:

AVERTISSEMENT

RISQUE D'EXPLOSION SI LA BATTERIE EST REMPLACÉE PAR UN MODÈLE INCORRECT. METTRE AU REBUT LES BATTERIES CONFORMÉMENT AUX INSTRUCTIONS.

Attention - Pour réduire les risques de chocs électriques et d'incendie

1. Cet équipement est conçu pour permettre la connexion entre le conducteur de mise à la terre du circuit électrique CC et l'équipement de mise à la terre. Voir les instructions d'installation.
2. Tout entretien sera entrepris par du personnel qualifié. Aucune pièce à l'intérieur de l'unité ne peut être remplacée ou réparée.
3. NE branchez pas, n'allumez pas ou n'essayez pas d'utiliser une unité manifestement endommagée.
4. Vérifiez que l'orifice de ventilation du châssis dans l'unité n'est PAS OBSTRUE.
5. Remplacez le fusible endommagé par un modèle similaire de même puissance, tel qu'indiqué sur l'étiquette de sécurité adjacente à l'arrivée électrique hébergeant le fusible.
6. Ne faites pas fonctionner l'appareil dans un endroit, où la température ambiante dépasse la valeur maximale autorisée. 40°C/104°F.
7. Débranchez le cordon électrique de la prise murale AVANT d'essayer de retirer et/ou de vérifier le fusible d'alimentation principal.

PRODUIT LASER DE CLASSE 1 ET RÉFÉRENCE AUX NORMES LASER LES PLUS RÉCENTES: IEC 60

825-1: 1993 + A1: 1997 + A2: 2001 ET EN 60825-1: 1994+A1: 1996+ A2: 2001

Unités à CA pour le Danemark, la Finlande, la Norvège, la Suède (indiqué sur le produit):

- Danemark - Unité de classe 1 - qui doit être utilisée avec un cordon CA compatible avec les déviations du Danemark. Le cordon inclut un conducteur de mise à la terre. L'unité sera branchée à une prise murale, mise à la terre. Les prises non-mises à la terre ne seront pas utilisées!
- Finlande (Étiquette et inscription dans le manuel) - Laite on liitettävä suojamaadoituskoskettimilla varustettuun pistorasiaan
- Norvège (Étiquette et inscription dans le manuel) - Apparatet må tilkoples jordet stikkontakt
- L'unité peut être connectée à un système électrique IT (en Norvège uniquement).
- Suède (Étiquette et inscription dans le manuel) - Apparaten skall anslutas till jordat uttag.

Pour brancher à l'alimentation électrique:

1. Branchez le câble d'alimentation à la prise principale, située sur le panneau arrière de l'unité.
2. Connectez le câble d'alimentation à la prise CA mise à la terre.

AVERTISSEMENT

Risque de choc électrique et danger énergétique. La déconnexion d'une source d'alimentation électrique ne débranche qu'un seul module électrique. Pour isoler complètement l'unité, débranchez toutes les sources d'alimentation électrique.

ATTENTION

Risque de choc et de danger électriques. Le débranchement d'une seule alimentation stabilisée ne débranche qu'un module "Alimentation Stabilisée". Pour Isoler complètement le module en cause, il faut débrancher toutes les alimentations stabilisées.

Attention: Pour Réduire Les Risques d'Électrocution et d'Incendie

1. Toutes les opérations d'entretien seront effectuées UNIQUEMENT par du personnel d'entretien qualifié. Aucun composant ne peut être entretenu ou remplacée par l'utilisateur.
2. NE PAS connecter, mettre sous tension ou essayer d'utiliser une unité visiblement défectueuse.
3. Assurez-vous que les ouvertures de ventilation du châssis NE SONT PAS OBSTRUÉES.
4. Remplacez un fusible qui a sauté SEULEMENT par un fusible du même type et de même capacité, comme indiqué sur l'étiquette de sécurité proche de l'entrée de l'alimentation qui contient le fusible.
5. NE PAS UTILISER l'équipement dans des locaux dont la température maximale dépasse 40 degrés Centigrades.
6. Assurez vous que le cordon d'alimentation a été déconnecté AVANT d'essayer de l'enlever et/ou vérifier le fusible de l'alimentation générale.

Sicherheitsanweisungen

VORSICHT

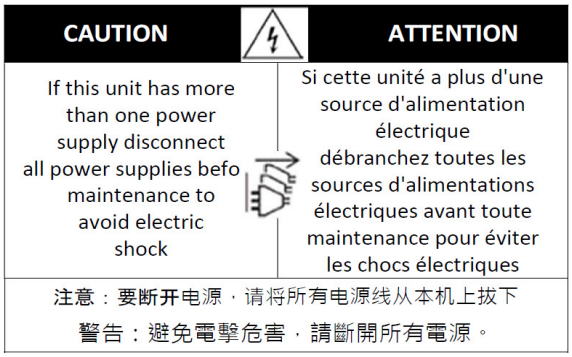
Die Elektroinstallation des Gebäudes muss ein unverzüglich zugängliches Stromunterbrechungsgerät integrieren.

Aufgrund des Stromschlagrisikos und der Energie-, mechanische und Feuergefahr dürfen Vorgänge, in deren Verlauf Abdeckungen entfernt oder Elemente ausgetauscht werden, ausschließlich von qualifiziertem Servicepersonal durchgeführt werden.

Zur Reduzierung der Feuer- und Stromschlaggefahr muss das Gerät vor der Entfernung der Abdeckung oder der Paneele von der Stromversorgung getrennt werden.

Folgende Abbildung zeigt das VORSICHT-Etikett, das auf die Radware-Plattformen mit Doppelspeisung angebracht ist.

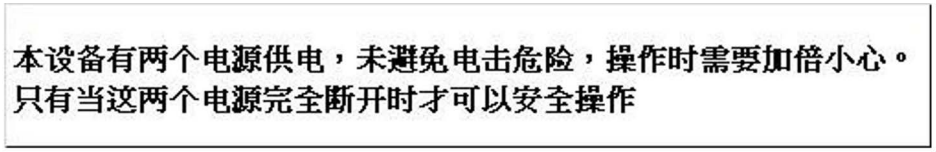
Figure 5: Warnetikett Stromschlaggefahr



SICHERHEITSHINWEIS IN CHINESISCHER SPRACHE FÜR SYSTEME MIT DOPPELSPEISUNG

Die folgende Abbildung ist die Warnung für Radware-Plattformen mit Doppelspeisung.

Figure 6: Sicherheitshinweis in chinesischer Sprache für Systeme mit Doppelspeisung



本設備有兩個電源供電，為避免電擊危險，操作時需要加倍小心。
只有當這兩個電源完全斷開時才可以安全操作。”

Übersetzung von Sicherheitshinweis in chinesischer Sprache für Systeme mit Doppelspeisung:

Die Einheit verfügt über mehr als eine Stromversorgungsquelle. Ziehen Sie zur Verhinderung von Stromschlag vor Wartungsarbeiten sämtliche Stromversorgungsleitungen ab.

WARTUNG

Führen Sie keinerlei Wartungsarbeiten aus, die nicht in der Betriebsanleitung angeführt sind, es sei denn, Sie sind dafür qualifiziert. Es gibt innerhalb des Gerätes keine wartungsfähigen Teile.

HOCHSPANNUNG

Jegliche Einstellungs-, Instandhaltungs- und Reparaturarbeiten am geöffneten Gerät unter Spannung müssen so weit wie möglich vermieden werden. Sind sie nicht vermeidbar, dürfen sie ausschließlich von qualifizierten Personen ausgeführt werden, die sich der Gefahr bewusst sind.

Innerhalb des Gerätes befindliche Kondensatoren können auch dann noch Ladung enthalten, wenn das Gerät von der Stromversorgung abgeschnitten wurde.

ERDUNG

Bevor das Gerät an die Stromversorgung angeschlossen wird, müssen die Schrauben der Erdungsleitung des Gerätes an die Erdung der Gebäudeverkabelung angeschlossen werden.

LASER

Dieses Gerät ist ein Laser-Produkt der Klasse 1 in Übereinstimmung mit IEC60825 - 1: 1993 + A1:1997 + A2:2001 Standard.

SICHERUNGEN

Vergewissern Sie sich, dass nur Sicherungen mit der erforderlichen Stromstärke und der angeführten Art verwendet werden. Die Verwendung reparierter Sicherungen sowie die Kurzschließung von Sicherungsfassungen muss vermieden werden. In Fällen, in denen wahrscheinlich ist, dass der von den Sicherungen gebotene Schutz beeinträchtigt ist, muss das Gerät abgeschaltet und gegen unbeabsichtigten Betrieb gesichert werden.

LEITUNGSSPANNUNG

Vor Anschluss dieses Gerätes an die Stromversorgung ist zu gewährleisten, dass die Spannung der Stromquelle den Anforderungen des Gerätes entspricht. Beachten Sie die technischen Angaben bezüglich der korrekten elektrischen Werte des Gerätes.

Plattformen mit 48 V DC verfügen über eine Eingangstoleranz von 36-72 V DC.

ÄNDERUNGEN DER TECHNISCHEN ANGABEN

Änderungen der technischen Spezifikationen bleiben vorbehalten.

Hinweis: Dieses Gerät wurde geprüft und entspricht den Beschränkungen von digitalen Geräten der Klasse 1 gemäß Teil 15B FCC-Vorschriften und EN55022 Klasse A, EN55024; EN 61000-3-2; EN; IEC 61000 4-2 to 4-6, IEC 61000 4-8 und IEC 61000-4- 11 für Konformität mit der CE-Bezeichnung. Diese Beschränkungen dienen dem angemessenen Schutz vor schädlichen Interferenzen bei Betrieb des Gerätes in kommerziellem Umfeld. Dieses Gerät erzeugt, verwendet und strahlt elektromagnetische Hochfrequenzstrahlung aus. Wird es nicht entsprechend den Anweisungen im Handbuch montiert und benutzt, könnte es mit dem Funkverkehr interferieren und ihn beeinträchtigen. Der Betrieb dieses Gerätes in Wohnbereichen wird höchstwahrscheinlich zu schädlichen Interferenzen führen. In einem solchen Fall wäre der Benutzer verpflichtet, diese Interferenzen auf eigene Kosten zu korrigieren.

BESONDERER HINWEIS FÜR BENUTZER IN NORDAMERIKA

Wählen Sie für den Netzstromanschluss in Nordamerika ein Stromkabel, das in der UL aufgeführt und CSA-zertifiziert ist 3 Leiter, [18 AWG], endend in einem gegossenen Stecker, für 125 V, [10 A changedMar12], mit einer Mindestlänge von 1,5 m [sechs Fuß], doch nicht länger als 4,5 m. Für europäische Anschlüsse verwenden Sie ein international harmonisiertes, mit "<HAR>" markiertes Stromkabel, mit 3 Leitern von mindestens 0,75 mm2, für 300 V, mit PVC-Umkleidung. Das Kabel muss in einem gegossenen Stecker für 250 V, 3 A enden.

BEREICH MIT EINGESCHRÄNKTEM ZUGANG

Das mit Gleichstrom betriebene Gerät darf nur in einem Bereich mit eingeschränktem Zugang montiert werden.

INSTALLATIONSCODES

Dieses Gerät muss gemäß der landesspezifischen elektrischen Codes montiert werden. In Nordamerika müssen Geräte entsprechend dem US National Electrical Code, Artikel 110 - 16, 110 - 17 und 110 - 18, sowie dem Canadian Electrical Code, Abschnitt 12, montiert werden. VERKOPPLUNG VON GERÄTEN Kabel für die Verbindung des Gerätes mit RS232- und Ethernet- müssen UL-zertifiziert und vom Typ DP-1 oder DP-2 sein. (Anmerkung: bei Aufenthalt in einem nicht-LPS-Stromkreis)

ÜBERSTROMSCHUTZ

Ein gut zugänglicher aufgeführter Überstromschutz mit Abzweigstromkreis und 15 A Stärke muss für jede Stromeingabe in der Gebäudeverkabelung integriert sein.

AUSTAUSCHBARE BATTERIEN

Wird ein Gerät mit einer austauschbaren Batterie geliefert und für diese Batterie durch einen falschen Batterietyp ersetzt, könnte dies zu einer Explosion führen. Dies trifft zu für manche Arten von Lithiumsbatterien zu, und das folgende gilt es zu beachten:

- Wird die Batterie in einem Bereich für Bediener eingesetzt, findet sich in der Nähe der Batterie eine Markierung oder Erklärung sowohl im Betriebshandbuch als auch in der Wartungsanleitung.
- Ist die Batterie an einer anderen Stelle im Gerät eingesetzt, findet sich in der Nähe der Batterie eine Markierung oder einer Erklärung in der Wartungsanleitung.

Diese Markierung oder Erklärung enthält den folgenden Warntext:

VORSICHT

EXPLOSIONSGEFAHR, FALLS BATTERIE DURCH EINEN FALSCHEN BATTERIETYP ERSETZT WIRD. GEBRAUCHTE BATTERIEN DEN ANWEISUNGEN ENTSPRECHEND ENTSORGEN.

- Denmark - "Unit is class I - mit Wechselstromkabel benutzen, dass für die Abweichungen in Dänemark eingestellt ist. Das Kabel ist mit einem Erdungsdraht versehen. Das Kabel wird in eine geerdete Wandsteckdose angeschlossen. Keine Steckdosen ohne Erdungsleitung verwenden!"
- Finland - (Markierungsetikett und im Handbuch) - Laite on liitettävä suojamaadoituskoskettimilla varustettuun pistorasiaan
- Norway - (Markierungsetikett und im Handbuch) - Apparatet må tilkoples jordet stikkontakt Ausschließlich für Anschluss an IT-Netzstromsysteme in Norwegen vorgesehen
- Sweden - (Markierungsetikett und im Handbuch) - Apparaten skall anslutas till jordat uttag.

Anschluss des Stromkabels:

1. Schließen Sie das Stromkabel an den Hauptanschluss auf der Rückseite des Gerätes an.
2. Schließen Sie das Stromkabel an den geerdeten Wechselstromanschluss an.

VORSICHT

Stromschlag- und Energiegefahr Die Trennung einer Stromquelle trennt nur ein Stromversorgungsmodul von der Stromversorgung. Um das Gerät komplett zu isolieren, muss es von der gesamten Stromversorgung getrennt werden.

Vorsicht - Zur Reduzierung der Stromschlag- und Feuergefahr

1. Dieses Gerät ist dazu ausgelegt, die Verbindung zwischen der geerdeten Leitung des Gleichstromkreises und dem Erdungsleiter des Gerätes zu ermöglichen. Siehe Montageanleitung.
2. Wartungsarbeiten jeglicher Art dürfen nur von qualifiziertem Servicepersonal ausgeführt werden. Es gibt innerhalb des Gerätes keine vom Benutzer zu wartenden Teile.
3. Versuchen Sie nicht, ein offensichtlich beschädigtes Gerät an den Stromkreis anzuschließen, einzuschalten oder zu betreiben.
4. Vergewissern Sie sich, dass sie Lüftungsöffnungen im Gehäuse des Gerätes NICHT BLOCKIERT SIND.
5. Ersetzen Sie eine durchgebrannte Sicherung ausschließlich mit dem selben Typ und von der selben Stärke, die auf dem Sicherheitsetikett angeführt sind, das sich neben dem Stromkabelanschluss, am Sicherungsgehäuse.
6. Betreiben Sie das Gerät nicht an einem Standort, an dem die Höchsttemperatur der Umgebung 40°C überschreitet.
7. Vergewissern Sie sich, das Stromkabel aus dem Wandstecker zu ziehen, BEVOR Sie die Hauptsicherung entfernen und/oder prüfen.

Electromagnetic-Interference Statements

The following statements are presented in English, French, and German.

Electromagnetic-Interference Statements

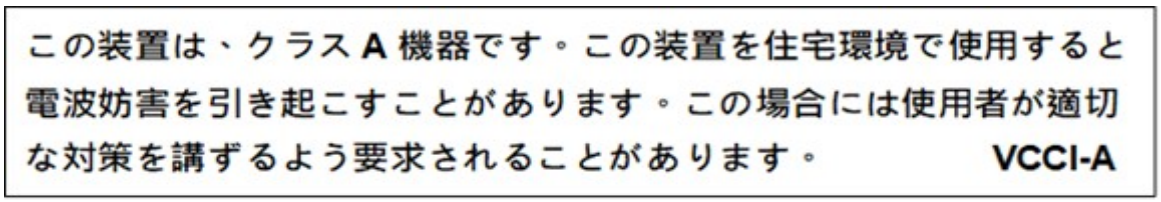
SPECIFICATION CHANGES

Specifications are subject to change without notice.

Note: This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15B of the FCC Rules and EN55022 Class A, EN 55024; EN 61000-3-2; EN 61000-3-3; IEC 61000 4-2 to 4-6, IEC 61000 4-8 and IEC 61000-4-11For CE MARK Compliance. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user is required to correct the interference at his own expense.

VCCI ELECTROMAGNETIC-INTERFERENCE STATEMENTS

Figure 7: Statement for Class A VCCI-certified Equipment



Translation of Statement for Class A VCCI-certified Equipment:

This is a Class A product based on the standard of the Voluntary Control Council for Interference by Information Technology Equipment (VCCI). If this equipment is used in a domestic environment, radio disturbance may occur, in which case, the user may be required to take corrective actions.

KCC KOREA

Figure 8: KCC—Korea Communications Commission Certificate of Broadcasting and Communication Equipment

Figure 9: Statement For Class A KCC-certified Equipment in Korean

이 기기는 업무용(A급) 전자파적합기기로서 판매자 또는 사용자는 이 점을 주의하시기 바라며, 가정외의 지역에서 사용하는 것을 목적으로 합니다.

HiddenComment: Added "Statement For Class A KCC-certified Equipment in Korean" per instruction from Yaniv Ben Dor 26 Jan. 2012, in SG_Changed KCC Guide.pdf. Note that copying the Korean text from the PDF into this FM doc was fine and distilled using print-to-file and distillation, but the built-in File>Print As PDF did not work. Here is the text in Korean from the PDF:

이 기기는 업무용 (A 급) 전자파적합기기로서 판
매자 또는 사용자는 이 점을 주의하시기 바라
며 , 가정외의 지역에서 사용하는 것을 목적으로
합니다 .

Translation of Statement For Class A KCC-certified Equipment in Korean:

This equipment is Industrial (Class A) electromagnetic wave suitability equipment and seller or user should take notice of it, and this equipment is to be used in the places except for home.

BSMI

Figure 10: Statement for Class A BSMI-certified Equipment

警告：為避免電磁干擾，本產品不應安裝或使用於住宅環境。

Translation of Statement for Class A BSMI-certified Equipment:

This is a Class A product, in use in a residential environment, it may cause radio interference in which case the user will be required to take adequate measures.

Déclarations sur les Interférences Électromagnétiques

MODIFICATIONS DES SPÉCIFICATIONS

Les spécifications sont sujettes à changement sans notice préalable.

Remarque: Cet équipement a été testé et déclaré conforme aux limites définies pour un appareil numérique de classe A, conformément au paragraphe 15B de la réglementation FCC et EN55022 Classe A, EN 55024, EN 61000-3-2; EN 61000-3-3; IEC 61000 4-2 to 4-6, IEC 61000 4-8, et IEC 61000-4-11, pour la marque de conformité de la CE. Ces limites sont fixées pour fournir une protection raisonnable contre les interférences nuisibles, lorsque l'équipement est utilisé dans un environnement commercial. Cet équipement génère, utilise et peut émettre des fréquences radio et, s'il n'est pas installé et utilisé conformément au manuel d'instructions, peut entraîner des interférences nuisibles aux communications radio. Le fonctionnement de cet équipement dans une zone résidentielle est susceptible de provoquer des interférences nuisibles, auquel cas l'utilisateur devra corriger le problème à ses propres frais.

DÉCLARATIONS SUR LES INTERFÉRENCES ÉLECTROMAGNÉTIQUES VCCI

Figure 11: Déclaration pour l'équipement de classe A certifié VCCI

この装置は、クラスA機器です。この装置を住宅環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。 VCCI-A

Traduction de la Déclaration pour l'équipement de classe A certifié VCCI:

Il s'agit d'un produit de classe A, basé sur la norme du Voluntary Control Council for Interference by Information Technology Equipment (VCCI). Si cet équipement est utilisé dans un environnement domestique, des perturbations radioélectriques sont susceptibles d'apparaître. Si tel est le cas, l'utilisateur sera tenu de prendre des mesures correctives.

KCC Corée

Figure 12: KCC—Certificat de la commission des communications de Corée pour les equipements de radiodiffusion et communication.

Figure 13: Déclaration pour l'équipement de classe A certifié KCC en langue coréenne

이 기기는 업무용(A급) 전자파적합기기로서 판매자 또는 사용자는 이 점을 주의하시기 바라며, 가정외의 지역에서 사용하는 것을 목적으로 합니다.

Translation de la Déclaration pour l'équipement de classe A certifié KCC en langue coréenne:

Cet équipement est un matériel (classe A) en adéquation aux ondes électromagnétiques et le vendeur ou l'utilisateur doit prendre cela en compte. Ce matériel est donc fait pour être utilisé ailleurs qu' á la maison.

BSMI

Figure 14: Déclaration pour l'équipement de classe A certifié BSMI

警告：為避免電磁干擾，本產品不應安裝或使用於住宅環境。

Translation de la Déclaration pour l'équipement de classe A certifié BSMI:

Il s'agit d'un produit de Classe A; utilisé dans un environnement résidentiel il peut provoquer des interférences, l'utilisateur devra alors prendre les mesures adéquates.

Erklärungen zu Elektromagnetischer Interferenz

ÄNDERUNGEN DER TECHNISCHEN ANGABEN

Änderungen der technischen Spezifikationen bleiben vorbehalten.

Hinweis: Dieses Gerät wurde geprüft und entspricht den Beschränkungen von digitalen Geräten der Klasse 1 gemäß Teil 15B FCC-Vorschriften und EN55022 Klasse A, EN55024; EN 61000-3-2; EN; IEC 61000 4-2 to 4-6, IEC 61000 4-8 und IEC 61000-4- 11 für Konformität mit der CE-Bezeichnung. Diese Beschränkungen dienen dem angemessenen Schutz vor schädlichen Interferenzen bei Betrieb des Gerätes in kommerziellem Umfeld. Dieses Gerät erzeugt, verwendet und strahlt elektromagnetische Hochfrequenzstrahlung aus. Wird es nicht entsprechend den Anweisungen im Handbuch montiert und benutzt, könnte es mit dem Funkverkehr interferieren und ihn beeinträchtigen. Der Betrieb dieses Gerätes in Wohnbereichen wird höchstwahrscheinlich zu schädlichen Interferenzen führen. In einem solchen Fall wäre der Benutzer verpflichtet, diese Interferenzen auf eigene Kosten zu korrigieren.

ERKLÄRUNG DER VCCI ZU ELEKTROMAGNETISCHER INTERFERENZ

Figure 15: Erklärung zu VCCI-zertifizierten Geräten der Klasse A

この装置は、クラスA機器です。この装置を住宅環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。 VCCI-A

Übersetzung von Erklärung zu VCCI-zertifizierten Geräten der Klasse A:

Dies ist ein Produkt der Klasse A gemäß den Normen des Voluntary Control Council for Interference by Information Technology Equipment (VCCI). Wird dieses Gerät in einem Wohnbereich benutzt, können elektromagnetische Störungen auftreten. In einem solchen Fall wäre der Benutzer verpflichtet, korrigierend einzugreifen.

KCC KOREA

Figure 16: KCC—Korea Communications Commission Zertifikat für Rundfunk-und Nachrichtentechnik

Figure 17: Erklärung zu KCC-zertifizierten Geräten der Klasse A

이 기기는 업무용(A급) 전자파적합기기로서 판매자 또는 사용자는 이 점을 주의하시기 바라며, 가정외의 지역에서 사용하는 것을 목적으로 합니다.

Übersetzung von Erklärung zu KCC-zertifizierten Geräten der Klasse A:

Verkäufer oder Nutzer sollten davon Kenntnis nehmen, daß dieses Gerät der Klasse A für industriell elektromagnetische Wellen geeignete Geräten angehört und dass diese Geräte nicht für den heimischen Gebrauch bestimmt sind.

BSMI

Figure 18: Erklärung zu BSMI-zertifizierten Geräten der Klasse A

警告：為避免電磁干擾，本產品不應安裝或使用於住宅環境。

Übersetzung von Erklärung zu BSMI-zertifizierten Geräten der Klasse A:

Dies ist ein Class A Produkt, bei Gebrauch in einer Wohnumgebung kann es zu Funkstörungen kommen, in diesem Fall ist der Benutzer verpflichtet, angemessene Maßnahmen zu ergreifen.

Altitude and Climate Warning

This warning only applies to The People's Republic of China.

对于在非热带气候条件下运行的设备而言，Tma：为制造商规范允许的最大环境温度，或者为 25°C，采用两者中的较大者。
关于在海拔不超过 2000m 或者在非热带气候地区使用的设备，附加警告要求如下：
关于在海拔不超过 2000m 的地区使用的设备，必须在随时可见的位置处粘贴包含如下内容或者类似用语的警告标记、或者附件 DD 中的符号。
“只可在海拔不超过 2000m 的位置使用。”

关于在非热带气候地区使用的设备，必须在随时可见的位置处粘贴包含如下内容的警告标记：

附件 DD：有关新安全警告标记的说明。
DD.1 海拔警告标记

标记含义：设备的评估仅基于 2000m 以下的海拔高度，因此设备只适用于该运行条件。如果在海拔超过 2000m 的位置使用设备，可能会存在某些安全隐患。
DD.2 气候警告标记

标记含义：设备的评估仅基于温带气候条件，因此设备只适用于该运行条件。如果在热带气候地区使用设备，可能会存在某些安全隐患。

Document Conventions

The following describes the conventions and symbols that this guide uses:

Item	Description	Description	Beschreibung
 **Example**	An example scenario	Un scénario d'exemple	Ein Beispielszenarium
 **Caution:**	Possible damage to equipment, software, or data	Endommagement possible de l'équipement, des données ou du logiciel	Mögliche Schäden an Gerät, Software oder Daten
 **Note:**	Additional information	Informations complémentaires	Zusätzliche Informationen
 **To**	A statement and instructions	Références et instructions	Eine Erklärung und Anweisungen
 **Tip:**	A suggestion or workaround	Une suggestion ou solution	Ein Vorschlag oder eine Umgehung
 **Warning:**	Possible physical harm to the operator	Blessure possible de l'opérateur	Verletzungsgefahr des Bedieners
 **IPv6 Ready**	Can use IPv6 (128-bit addresses) as well as IPv4 (32-bit addresses)	Peut utiliser IPv6 (adresses 128-bit,) ainsi que IPv4 (adresses 32-bit)	Kann sowohl IPv6 (128-Bit Adressen) als auch IPv4 (32-Bit Adressen) verwenden

TABLE OF CONTENTS

CHAPTER 1 – INTRODUCING APPSHAPE++

This section includes the following topics:

AppShape++ Overview

AppShape++ is a framework for customizing application delivery using user-written scripts.

AppShape++ provides the flexibility to control application flows and fully meet business requirements in a fast and agile manner.

The AppShape++ framework enables you to:

- Extend ADC Fabric services with delivery of new applications
- Quickly deploy new services
- Mitigate application problems without changing the application
- Preserve infrastructure investment by adding new capabilities without additional equipment investment

AppShape++ provides specific API extension to the Tool Command Language (Tcl) to query and manipulate data, and take actions such as server selection. The AppShape++ scripts can be attached to virtual service thus allowing to perform protocol content switching decisions and modification on any TCP/UDP protocol.

Tool Command Language (Tcl)

Tcl is a powerful open source, dynamic programming language, suitable for a wide range of uses. Tcl is a cross-platform language, which is easily deployed and highly extensible. For more information about Tcl and Tk, its GUI developing extension, see www.tcl.tk/.

For a Tcl tutorial, see www.tcl.tk/man/tcl/tutorial/tcltutorial.html.

For the Tcl command reference, see www.tcl.tk/man/tcl8.4/TclCmd/contents.htm.

For the Tclers wiki, see http://wiki.tcl.tk/.

AppShape++ Tcl Support

AppShape++ is built over Tcl version 8.5.9. AppShape++ provides an API extension to the Tcl commands. Some Tcl commands and the interface to library files are disabled in AppShape++. For the supported Tcl commands, see Appendix A - Supported Tcl Commands, page 215.

AppShape++ Scripting Principles

This section describes the basic elements of AppShape++ scripting, and describes the following topics:

Event-driven Scripts

AppShape++ scripts are event-driven—the flow of the script is determined by events such as TCP connection accepted, HTTP request received, and the handling of each event.

AppShape++ provides a number of events specific to the Alteon environment, including TCP/UDP connection events, and HTTP transaction events. For the full list of events, see AppShape++ Events, page 205.

Event handling is implemented using Tcl commands and AppShape++ command extensions. For a full list of AppShape++ commands, see AppShape++ Commands, page 45.

To implement event selection in the script, AppShape++ introduces the `when` command extension. The syntax of event selection and handling is `when <event> {code}`. For example:

```
when HTTP_REQUEST {
HTTP::header insert “X-Forwarded-For” [IP::client_addr]
}
```

Script Structure

An AppShape++ script includes one or more of the following sections:

- Declaration Section, page 39
- Initialization Section, page 39
- Main Section, page 40

Declaration Section

The declaration section is optional, but when it exists it must appear at the beginning of the script, before any other command or event.

Currently this is used to declare server groups that are addressed in the script using the `group` command. This is required because:

- The `group` command can only address (select or retrieve status) server instances that are available for the virtual service on which the script runs.
- The server instances available for a virtual service are created for the server groups attached to the virtual service via Alteon configuration based on the virtual service service port (vport), virtual service server port (rport), and real server ports (addport).
- Server instances are not automatically created for server groups that are not attached to a virtual service via Alteon configuration, but are addressed in the script attached to the virtual service. To create the server instances, the group must be specifically declared as active for the virtual service using the `attach group` command in the script.

This is an example of a declaration section:

```
attach group 5
attach group 17
attach group 214
```

Initialization Section

The initialization section is used to declare global static variables. Static variables initialized in one script can be accessed (read-only) by all other scripts.

The initialization section is implemented using the INIT event. Within this event you can use Tcl commands and the AppShape++ log command.

The INIT event occurs only when a script is created or updated (after the `Apply` command).

This is an example of an initialization section:

```
when INIT {
set static::STATUS_CODE "200"
set static::CONTENT "Apologies, service unavailable"
}
```

Main Section

The main section includes the different events you want to catch, and how each event should be handled.

This is an example of a main section:

```
when HTTP_REQUEST {
if { [group count active_servers 1] == 0 }
HTTP::respond
$static::STATUS_CODE content
$static::CONTENT
}
```

AppShape++ Variables

AppShape++ supports the following types of variables:

- **Global static variables**—These variables are declared only in the INIT area of a script, and can be addressed from all scripts (using the variable command).
- **Local dynamic variables**—These variables are declared inside the script main area, and are valid only for that script and only per connection (the same variable can hold different values for different connections processed by the same script).

Script Validation

The validation process takes place when a script is created or updated (after the **Apply** command) and checks that

- Commands are recognized
- Commands are valid in the context (event)
- Sub-commands and the number of arguments (per sub-command) are correct
- All variables are defined or declared
- Local variables are not used in INIT events
- Static variables are not modified outside INIT events
- Best-effort numeric argument validation (for constant arguments only) is used

Notes

- The validation mechanism does not guarantee that the script is valid, but it helps to identify the most common errors.

- Only tokens known at compile time are validated.

Troubleshooting

Several tools are available to troubleshoot AppShape++ scripts:

- Compilation errors are reported during the Apply process. For more information, see Script Validation, page 40.
- When run-time errors occur, the connection is closed and a message is sent to the Application Services Log (applog).

  The following run-time errors can occur

  — Invalid command (for example, trying to replace content when the offset is invalid).

  — Zero division.

  — Time limit exceeded.

  — Error initiated by the script itself with an “error” command.

  Note: If a script fails, scripts of a lower priority will not run.
- The AppShape++ log command can be used in a script to send a troubleshooting message to the applog. To log these messages, AppShape++ logging to the applog must be enabled.
- The Tcl `catch` command can be used in scripts to handle recovery from an error without ending the connection.

  In the following example, the addressed group has been removed from the configuration:

```
when HTTP_REQUEST {
if { [catch {group select 200 }] } {
log “group ID 200 does not exist”
}
}
```

- Statistics are available for each script, and for each event within each script. The statistics count the number of times that a script or script event was activated, failed or aborted.
- The regex option does not work in switch commands when the regular expression contains square brackets [].

Best Practice Guidelines

This section describes guidelines for the following AppShape++ script coding practices:

- Encoding Support, page 41
- Performance Optimization, page 42

Encoding Support

When processing HTML pages or other textual protocols that may contain non-ASCII characters, use the `HTTP:payload find` command instead of a string command on `HTTP::payload` (or `TCP:payload find`/`UDP::payload find`, depending on protocol).

This is because character length and byte length of the payload are not necessarily the same.

For example, the following script could be used to insert the word "Hello" at the end of an HTML title:

```
when HTTP_RESPONSE {
HTTP::collect
}
when HTTP_RESPONSE_DATA {
set payload [string tolower [HTTP::payload]]
set offset [string first </title> $payload 0]
HTTP::payload replace $offset 0 "Hello"
}
```

If the title is non-ASCII, the script will not work, since the `string first` command returns the character offset, while the byte offset is bigger (because non-ASCII characters have a longer Unicode representation).

To avoid this problem, use `HTTP::payload find` instead of `string first`.

```
when HTTP_RESPONSE {
HTTP::collect
}
when HTTP_RESPONSE_DATA {
set byte_offset [HTTP::payload find –nocase {</title>} ]
HTTP::payload replace $byte_offset 0 "Hello"
}
```

Performance Optimization

The following recommendations can improve script performance:

Payload Processing

- Collecting the payload is a resource-consuming operation. Use only when needed, and collect only the length needed.
- The `HTTP/TCP/UDP::payload find` command searches for content in the payload faster than the native Tcl string `first command`.

Variable Assignment

- Avoid assigning a command result to a variable. For example, if you need to compare URI once, use `[HTTP::uri]` inside the compare function.

  However, if the command result needs to be used several times in a script, you should assign it to a variable. For example, if you need the URI several times, or from several events, use `set uri [HTTP::uri]` and then use the local variable.
- When assigning a string to a variable, if no substitution is needed inside, use {}.

String Searches

The regular expression search (regexp) is very flexible but highly resource-consuming. While regular string search is also resource-intensive, a regexp search may be use fewer resources than several string searches.

Miscellaneous

- When using the Tcl `expr` command, always use {}, even with variables inside.
- For initializations of static data that is not traffic-dependent, always use the INIT section and static variables.

AppShape++ Script Management

To implement AppShape++ scripts on Alteon

1. Develop a script.

   You can develop a script with any text editor, or write directly in the Alteon browser-based interface.

2. Install the script on Alteon.

   You can import a script as a file, or paste a script directly into the Alteon browser-based interface or CLI.

   When you execute the **Apply** command, new or changed scripts are compiled and compilation errors are reported.

3. Attach the script to a virtual service or to a filter.

   You can attach a script to any number of virtual services or filters.

   You can attach up to 16 scripts to the same service or filter. Alteon executes the scripts in the order determined by the configured priority of each script.

For more information about importing and exporting AppShape++ scripts, see *AppShape++ Scripting* in the *Alteon Web Based Management Application Guide* or in the *Alteon Command Line Interface Application Guide*.

CHAPTER 2 – APPSHAPE++ COMMANDS

This section describes the commands introduced by AppShape++ as extensions to Tcl commands. The commands are divided into these categories:

Declaration Commands

Declaration commands declare configuration elements as available for the script.

This section describes the following declaration command:

attach

Description

Declares configuration objects that need to be used by the AppShape++ script in a manner different than their configuration.

Implementation Notes

All attach commands must appear at the beginning of the script, before any other command or event. Attached commands are not part of any event.

Example:

```
#Group selection script
attach group 2
attach group 13
attach group 21
when INIT {.....
```

First Implemented Version

29.0.0

Sub-commands

- group
- snat

group

Description	Declares groups that need to be available for load balancing to the virtual service to which the AppShape++ script is attached. Required only when the script performs group or server selection. All groups that are addressed in the script (group command) must be declared, so that Alteon can prepare appropriate server instances available for the virtual service, otherwise the group command can fail.
Input	Group ID
Full Syntax	`attach group <groupID>`

snat

Description	Declares IP addresses used to translate the source IP. All IP addresses used in the script to perform NAT (snat command) must be declared, so that the device can respond to ARP messages for these IP addresses, and ensure responses to these IP addresses reach Alteon.
Input	IP address
Full Syntax	`attach snat <IP address>`

Global Commands

Global commands are generic commands that can be applied to all traffic types (TCP, UDP, and HTTP).

This section describes the following global commands:

b64encode

Description
Encodes a string in base64.

Input
A regular string.

Output
A base64-encoded string.

Full Syntax
```
b64encode <orig_string>
```

Valid Events
All

First Implemented Version
30.0.0

b64decode

Description
Decodes a string in base64.

Input
A base64-encoded string.

Output
A decoded string.

Full Syntax
```
b64decode <orig_string>
```

Valid Events
All

First Implemented Version
30.0.0

class

Description

Provides access to data classes. Includes methods for searches for a specific name, and iterative searches.

Valid Events

All traffic events

First Implemented Version

29.3.0

Sub-commands

- exists
- type
- size
- match
- search
- startsearch
- nextelement
- anymore
- apmlist

exists

Description	Checks if the specified data class exists.
Input	Data class ID.
Output	1 (for true) and 0 (for false).
Full Syntax	`class exists <data class id>`

type

Description	Retrieves the type of data in the specified data class. The supported types are integer, string and IP address.
Input	Data class ID.
Output	The type of data class.
Full Syntax	`class type <data class id>`

size

Description	Retrieves the number of entries in the specified data class.
Input	Data class ID.

Output	Integer
Full Syntax	`class size <data class id>`

match

Description	Checks if an argument matches one of the names in the specified data class. The matching criteria is defined by the operator used, and the output is determined by the option used.
Options	-name: returns the name of the matched element, or null if none matched. -value: returns the value of the matched element, or null if none matched or no value exists. -all: returns multiple matches, if found. This option must be used with the name or value option
Operators	equals, contains, starts_with, ends_with
Input	Argument Data class ID
Output	By default 1 (for match) or 0 (for no match). If one of the options is used, the output changes accordingly.
Full Syntax	`class match [<options>] <argument> <operator> <data class id>`
Notes	If an operator finds multiple matches, it returns the first match.

search

Description	Checks if any of the names in the specified data class matches the argument. The matching criteria is defined by the operator used, and the output is determined by the option used.
Options	-name: returns the name of the matched element, or null if none matched. -value: returns the value of the matched element, or null if none matched or no value exists. -all: returns a list containing the name and value of the matched element, or {-1, null, null} if none matched.
Operators	equals, contains, starts_with, ends_with
Input	Argument Data class ID
Output	By default 1 (for match) or 0 (for no match). If one of the options is used, the output changes accordingly.
Full Syntax	`class search [<options>] <data class id> <operator> <argument>`

startsearch

Description	Initializes the index for an iterative search of the specified data class.
Input	Data class ID
Full Syntax	`class startsearch <data class id>`

nextelement

Description	Retrieves the next element in the specified data class.
Options	-name: returns the name of the matched element, or null if none matched. -value: returns the value of the matched element, or null if none matched or no value exists.
Input	Index Data class ID
Output	The name and value of the next element (as a list).
Full Syntax	`class nextelement [<options>] <data class id>`

anymore

Description	Checks if there are any more elements in the data class when performing an iterative search.
Input	Data class ID
Output	1 (for true) and 0 (for false).
Full Syntax	`class anymore <data class id>`

apmlist

Description	Globally sets a data class as an APM exclusion list. The data class should contain value pairs of <URL, "0"|"1">, where 0 is for APM exclusion, and 1 sets an alternate insertion point for APM.
Options	-regex: all data class values are matched as regular expressions.
Input	Data class ID
Output	None
Full Syntax	`class apmlist [<-regex>] <data class id>`
Notes	For use only in "APM_script", [init] events.

cpu usage

Description
Retrieves the CPU utilization average for the current SP core for the last 1, 4, or 64 seconds.

Full Syntax
```
cpu usage [ 1 | 4 | 64]
```

Valid Events
All

First Implemented Version
30.5.0

deflate

Description
Compress specified content using DEFLATE algorithm.

Input
string

Output
string

Full Syntax
deflate <content>

Valid Events
All

First Implemented Version
34.5

event

Description
Provides control over the execution of AppShape++ scripts for the current session.

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED

SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version

29.0.0

Sub-commands

- name
- disable_current
- disable
- enable
- disable_all
- enable_all

name

Description	Returns the name of the current event.
Output	The name of the current event as it appears after the “when” keyword.
Full Syntax	`event name`

disable_current

Description	Stops execution of the current event after the event code section is finished by skipping the execution of following scripts for the current event. Does not affect the next execution of this type of event on this connection.
Full Syntax	`event disable_current`

disable

Description	Disables the execution of specified event types for the rest of the connection (unless re-enabled).
Input	Event names (space-separated).
Full Syntax	`event disable <event name>+`

enable

Description	Re-enables the execution of specified event types.
Input	Event names (space-separated).
Full Syntax	`event enable <event name>+`

disable_all

Description	Disables the execution of all AppShape++ events for the rest of the connection (unless re-enabled).
Full Syntax	`event disable_all`

enable_all

Description	Re-enables the execution of all AppShape++ events that were disabled previously on this connection.
Full Syntax	`event enable_all`

filter

Description

Provides access to filter properties and allows Alteon to skip a filter.

Valid Events

CLIENT_ACCEPTED

HTTP_REQUEST

HTTP_REQUEST_DATA

Implementation Notes

All traffic events for informational sub-commands are applicable.

First Implemented Version

29.3.0

Sub-commands

- name
- id
- skip

name

Description	Returns the name of the currently matched filter, as defined in the configuration. Returns an empty string if no name is configured, or no selection has been made.
Output	The name of the current filter.
Full Syntax	`filter name`

id

Description	Returns the ID of the currently matched filter. Returns an empty string if no selection has been made.
Output	The ID of the current filter.
Full Syntax	`filter id`

skip

Description	Ignores the currently matched filter and continues to look for a filter matching the connection.
Full Syntax	`filter skip`

get_session_id

Description
Returns Alteon proxy internal session id number.

Output
Integer

Full Syntax
get_session_id

Valid Events
All

First Implemented Version
33.0.2.0

group

Description
Accesses a server group, and lets you retrieve information about a group server.

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version
29.0.0

Sub-commands
- select

- count
- list

select

Description	Selects the specified group and, optionally, the specific real server belonging to the group, to which the command forwards the current session. To be able to select a group for a virtual service, that group must be declared as active for that service, either via configuration or script declaration.
Options	• server • all—Allows the same request to be broadcast to all the servers in the group. This option is available only in virtual services and only for UDP traffic.
Input	group ID server ID or server IP (optional) port (optional)
Full Syntax	`group select <group ID> [all | server <server ID | IP> [<port>]]`
Alias	pool <group ID> member <server ID>)

count

Description	Retrieves the total number of configured servers (or only the number of configured servers that are active) in the specified group that are available for load balancing for the current service.
Output	The number of servers (integer).
Full Syntax	`group count all_servers | active_servers <group ID>`
Alias	active_members <group ID> alias for group count active_servers <group ID> members <group ID> alias for group count all_servers <group ID>

list

Description	Retrieves all the servers configured (or only the configured servers that are active) for the specified group, in Tcl list format.
Output	The list of servers.
Full Syntax	`group list all_servers | active_servers <group ID>`

hex

Description
Transforms text string into hex string.

Options
-f for "wireshark-like" output

Input
Text string

Output
Hex string

Full Syntax
```
hex
```

Valid Events
All

First Implemented Version
33.5.2.0, 33.0.6.0

host

Description
Overrides the server selection mechanism and forwards the request to a specified host (IP and, optionally, port) that is not defined as a real server on Alteon.

Input
IP address

Port (optional)

Full Syntax
```
host <IP> [<port>]
```

Valid Events
All except PERSIST_DOWN

First Implemented Version
30.1.0

Implementation Notes
- If the port is not specified, traffic is sent to the initial destination port used by the client.

- For HTTP filters with *Parse All* enabled, do not use this command in CLIENT_ACCEPTED event (implemented in version 33.0.2.0).
- The IP version of the IP used in host command must fulfill the following:
  - When used in a virtual service, it must match the virtual server IP version
  - When used in an Allow filter, it must match the IP version of the filter
  - When used in a Redirect filter, it must match the IP version of the associated group
- The host command can be used to perform IPv4/v6 gateway when the host IP version differs from the IP version of the original destination IP with the following caveats:
  - On virtual services it is supported starting with versions 33.5.8.0 and 34.0.4.0.
  - On filters it is supported for Redirect filter and has the following requirements:
    - The host IP version must be the same as the IP version of the group attached to the filter.
    - The destination MAC address must be the same as the MAC address of a server in the group (usually the server would be the next hop router).

inflate

Description

Uncompress specified content using DEFLATE algorithm.

Input

string

Output

string

Full Syntax

inflate <content>

Valid Events

All

First Implemented Version

33.5

log

Description

Logs a specified message to the Alteon application services log and/or sends as a syslog message. Also requires the following steps:

- AppShape++ logging is globally enabled on Alteon.
- The syslog messages are sent via the data port.
- The syslog messages are sent to the syslog hosts defined at `/cfg/sys/syslog/hst`.
- The syslog messages are sent only over UDP port 514.

Input

The specified message.

The syslog facility (optional).

The severity level (optional).

Output

-s sends a message to the syslog server only.

-a sends a message to both the syslog and applog servers.

Full Syntax

```
log [-s |-a] [facility] [debug | info | warning | error] <message>
```

Valid Events

INIT

CLIENT_ACCEPTED

CLIENT_DATA

CLIENT_CLOSED

SERVER_CONNECTED

SERVER_DATA

SERVER_CLOSED

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP_RESPONSE

HTTP_RESPONSE_DATA

HTTP_RESPONSE_CONTINUE

HTTP_CACHE_RESPONSE

Implementation Notes

- The message can contain variables (that are replaced by a variable value), or command output (such as an IP address).
- By default the command logs the message only in the application services log.

First Implemented Version
29.0.0

md5

Description
Calculates the md5 value

Input
Text

Output
Hash value (32 bytes)

Full Syntax
```
md5 <text>
```
For example:
```
md5 [HTTP::uri]
```

Valid Events
All events

First Implemented Version
29.4.0

persist

Description
Enables implementing persistent connections and manipulating the persistence table.

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA

HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

Implementation Notes

This command can be used only when the selected group is known.

First Implemented Version

29.4.0

Sub-commands

- usid
- cookie
- source_addr
- dest_addr
- hash
- none
- add
- lookup
- delete
- timeout

usid

Description	Implements persistence by user-defined session ID.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group.
Input	session ID (string) timeout (optional)
Full Syntax	`persist usid <session id> [any virtual|any service|any group|group <group>][<timeout>]`
Alias	persist uie
Valid Events	All client side events that run before LB_SELECTED.
Notes	This command searches the persistence table for an entry that matches the provided session ID string, the current virtual service, and the selected group ID. If found, traffic is forwarded to the server mentioned in the persistence entry. If not found, an entry is created for future requests belonging to that session. The entry creation must take place after the server is selected.

cookie

Description	Implements persistence for HTTP requests by specified cookie value, using either the cookie insert, cookie rewrite, or passive cookie mechanism.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group.
Input	cookie name expires (optional) offset (optional) length (optional) timeout (optional)

Full Syntax	`persist cookie insert <cookie name> [expires <expires> [absolute | relative] ] [path <path>] [domain <domain>] [secure <enable | disable>] [httponly <enable | disable>] [samesite <lax | strict | none>] [any virtual|any service|any group|group <group>] [timeout <timeout>]`
	`persist cookie rewrite <cookie name> [expires<expires> [absolute | relative] ] [path <path>] [domain <domain>] [secure <enable | disable>] [httponly <enable | disable>] [samesite <lax | strict | none>] [any virtual|any service|any group|group <group>] [timeout <timeout>]`
	`persist cookie passive <cookie name> [{<offset> [<length>]}] [any virtual|any service|any group|group <group>] [timeout <timeout>]`

Valid Events	HTTP_REQUEST HTTP_REQUEST_DATA HTTP_CRULE_MATCH HTTP_CRULE_NOMATCH

Notes	**Cookie Insert** The cookie insert command searches for a specified cookie value in the HTTP request. If found, the command retrieves the relevant server from the persistence table. If not found, the command selects a server according to group metrics and on server response. A cookie with the specified name is inserted in the HTTP response, and a persistence table entry is added. When no expires value is specified, or the value is 0, the expires parameter is not included in the Set-Cookie header. Radware recommends that in such cases you specify a timeout value. The default timeout value is 10 minutes (600 seconds). When the expires value is not 0, an expires parameter is included in the Set-Cookie header. In this case, the timeout option is not valid.
	Cookie Passive The cookie passive command searches for a specified cookie value. If found, the command retrieves the relevant server from the persistence table. If not found, the command selects a server according to group metrics and on server response. A cookie with the specified value is retrieved from the HTTP response, and a persistence table entry is added. When the server cookie includes an expires parameter, its value sets the persistence entry lifetime. If such a parameter does not exist and the timeout option is specified, the timeout value is used as the persistence entry idle timeout (default 10 minutes).
Notes (continued)	**Cookie Rewrite** The cookie rewrite command searches for a specified cookie value. If found, the command retrieves the relevant server from the persistence table. If not found, the command selects a server according to group metrics and on server response. A cookie with an Alteon-generated value is inserted in the HTTP response, and a persistence table entry is added. When the expires option is not specified: • If an expires parameter exists in the server cookie, the parameter is preserved and its value sets the persistence entry lifetime. If the timeout option is specified, it is ignored. • If no expires parameter exists in the server cookie, the timeout option value, if present, is used as the persistence entry idle timeout (default 10 minutes). When the expires option value is 0, the expires parameter is removed from the server cookie (if it exists). Radware recommends that in such cases, you specify a timeout value. The default timeout value is 10 minutes (600 seconds). When the expires option value is not 0, the expires parameter is included in the server cookie, or if it already exists, its value is overwritten. In this case, the timeout parameter is not valid.
	Expire and Timeout The time that a persistence entry exists can be specified in terms of lifetime (entry is deleted X seconds after it was created), or timeout (entry is deleted Y seconds after the last time the persistence entry was looked up). When the expires parameter is used or exists (in passive and rewrite server cookies), it defines an entry with lifetime. When a timeout value is used, it creates an entry with timeout.

source_addr

Description	Implements persistence for a source IP address.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group.
Input	source IP address timeout (optional)
Full Syntax	`persist source_addr <IPv4 mask> | <IPv6 prefix> [any virtual | any service | any group | group <group>] [<timeout>]`
Valid Events	All client-side events that run before LB_SELECTED.
Notes	The command searches the persistence table for an entry that matches the provided session ID string, the current virtual service, and the selected group ID. If found, traffic is forwarded to the server mentioned in the persistence entry. If not found, an entry is created for future requests belonging to that session. The entry creation must take place after the server is selected.

dest_addr

Description	Implements persistence for a destination IP address.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group.
Input	destination IP address timeout (optional)
Full Syntax	`persist dest_addr <IPv4 mask> | <IPv6 prefix> [any virtual | any service | any group | group <group>] [<timeout>]`
Valid Events	All client-side events that run before LB_SELECTED.
Notes	The command searches the persistence table for an entry that matches the packet destination address subnet (taking into consideration the provided subnet mask/prefix), the current virtual service, and the selected group ID. If found, traffic is forwarded to the server mentioned in the persistence entry. If not found, an entry is created for future requests belonging to that session. The entry creation must take place after the server is selected.

hash

Description	Implements persistence by performing hashing on the specified session ID string.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group.
Input	session ID timeout (optional)
Full Syntax	`persist hash <session id> [any virtual | any service | any group | group <group>] [<timeout>]`
Valid Events	All client-side events that run before LB_SELECTED.
Notes	The command performs hashing on the provided string, and searches the persistence table for an entry that matches the hash result, the current virtual service and the selected group ID. If found, traffic is forwarded to the server mentioned in the persistence entry. If not found, an entry is created for future requests belonging to that session. The entry creation must take place after the server is selected. Since hashing is performed on the session ID, multiple sessions result in the same hash value and use the same server.

none

Description	Disables persistence for the current session.
Full Syntax	`persist none`
Valid Events	All client-side events that run before LB_SELECTED.

add

Description	Adds a new persistence entry with the specified parameters and the currently selected server. Valid in server-side events.
Options	• Any group—Adds a session ID entry that can be used for all groups of the current virtual service. • Any service—Adds a session ID entry that can be used for all services of the current virtual server. • Any virtual—Adds a session ID entry that can be used for all virtual servers. • Group—Adds a session ID entry for the specified group. • -o—Outbound

Input	mode (usid, hash, source_addr, dest_addr) value timeout (optional) group id (optional)
Full Syntax	`persist add <mode> <value> | {<value> any virtual | any service | any group | group <group>}] [-o [-gid <group id>] [-pr <port1,port2-port3>] ] [<timeout>]`
Alias	• without -o—Server-side events without HTTP_CACHE_RESPONSE. LB_SELECTED, LB_FAILED. • with -o—All transport events without HTTP_CACHE_RESPONSE.
Notes	This command can be used only when group and server are already selected.

lookup

Description	Searches for the specified session entry, and returns the selected server.
Options	• All—Retrieves the group ID, server, and port. • Server—Retrieves the server ID. • Port—Retrieves the server port. • Group—Retrieves the group ID. • Any group—Retrieves the session ID for the current virtual service, irrespective of the group ID to which the entry belongs. • Any service—Retrieves the session ID for the current virtual server, irrespective of the service and group ID to which the entry belongs. • Any virtual—Retrieves the session ID irrespective of any virtual service and group parameter. • Group—Retrieves the session ID for the entries for the current virtual service and specified group.
Input	mode (usid, hash, source_addr, dest_addr) value timeout (optional) group id (optional)
Output	Depending on the output option specified, one of the following: • Group ID, server, and port • Server only • Port only • Group ID only
Full Syntax	`persist lookup <mode> <value> | {<value> any virtual | any service | any group | group <group>}] [all | server | port | group]`
Valid Events	All transport events without HTTP_CACHE_RESPONSE.

delete

Description	Deletes the specified session entry.
Options	• Any group—Retrieves the session ID for the current virtual service, irrespective of the group ID to which the entry belongs. • Any service—Retrieves the session ID for the current virtual server, irrespective of the service and group ID to which the entry belongs. • Any virtual—Retrieves the session ID irrespective of any virtual service and group parameter. • Group—Retrieves the session ID for the entries for the current virtual service and specified group.
Input	• mode (usid, hash, source_addr, dest_addr) • value • group id (optional) • timeout
Full Syntax	`persist delete <mode> <value> | {<value> any virtual | any service | any group | group <group>}]`
Valid Events	All transport events without HTTP_CACHE_RESPONSE.

timeout

Description	Updates the timeout of the specified persistence entry.
Options	• notouch—When specified, the entry timestamp must not be updated. • Any group—Retrieves the session ID for the current virtual service, irrespective of the group ID to which the entry belongs. • Any service—Retrieves the session ID for the current virtual server, irrespective of the service and group ID to which the entry belongs. • Any virtual—Retrieves the session ID irrespective of any virtual service and group parameter. • Group—Retrieves the session ID for the entries for the current virtual service and specified group.
Input	• mode (usid, hash, source_addr, dest_addr) • value • group id (optional) • timeout
Output	Timeout is updated if update succeeds; otherwise 0.
Full Syntax	`persist timeout [-notouch] <mode> <value> [any virtual | any service | any group | group <group>] <timeout>`
Valid Events	All transport events without HTTP_CACHE_RESPONSE.

service

Description

Provides access to service properties.

Input

The specified message.

Full Syntax

```
log [debug | info | warning | error] <message>
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version

29.0.0

Sub-commands

- name
- ip
- port

name

Description	Returns the name of the current virtual service as defined in the configuration. If no name is configured, returns an empty string.
Output	The name of the current virtual service (VIP) in string format.
Full Syntax	`service name`

ip

Description	Returns the IP address of the current virtual service (VIP) in string format.
Output	The IP address of the current virtual service.
Full Syntax	`service ip`

port

Description	Returns the service port of the current virtual service.
Output	The port of the current virtual service (integer).
Full Syntax	`service port`

snat

Description

Performs source network address translation (NAT) with the specified address on the connection, or disables source NAT for the connection.

Input

IP address

Port (optional)

Implementation Notes

This command overwrites the group and real server PIP address configuration, but does not overwrite the disabled PIP mode.

This command cannot be performed if PIP mode is disabled for the virtual service or the selected real server.

This command is supported only in redirect and outbound LLB filters.

First Implemented Version

30.2.0

Sub-commands

- snat

snat

Description	Performs source network address translation (NAT) with the specified address on the connection, or disables the source NAT for the connection.
Input	IP address

Full Syntax	`snat <address> [is_dynamic] | none`
Notes	The NAT address must be declared using the attach command. The `is_dynamic` flag is used for addresses that are not attached to the script.

table

Description

Provides access to generic persistent memory. The generic persistent memory can be used to provide storage for dynamic variables that must be shared between services and sessions.

First Implemented Version

29.4.0

Sub-commands

- add
- append
- delete
- increment
- lookup
- replace
- set
- timeout

add

Description	Creates a new entry in the table. Returns an error if the entry already exists.
Options	-subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key value timeout (default value is 600 seconds, use "indefinite" for entries without expiration) subtable name (optional)
Full Syntax	`table add [-subtable <name>] [--] <key> <value> [<timeout>]`
Notes	This command is a private case of table set command with flag -excl present

append

Description	Appends a new string to the value already present in the table. If the entry does not exist in the table, the command creates a new entry with the value set to the appended string. The original value is an empty string.
Options	-mustexist—When specified, no append is performed and the command returns a null string if the table entry does not exist. -subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key append value subtable name (optional)
Output	new value
Full Syntax	`table append [-notouch] [-subtable <name>] [-mustexist] [--] <key> < append value> [<timeout>]`

delete

Description	Deletes the specified entry in the table or subtable.
Input	key subtable name (optional)
Output	new value
Full Syntax	`table delete [-subtable <name>] [--] <key>`

increment

Description	Adds a specified increment to the value of the entry. If the entry does not exist in the table, the command creates a new entry with the value set to the increment value.
Options	-notouch—When specified, the entry timestamp must not be updated. -mustexist—When specified, no increment is performed and the command returns a null string if the table entry does not exist. -subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key increment subtable name (optional)
Output	new value (after increment)
Full Syntax	`table increment [-notouch] [-subtable <name>] [-mustexist] [--] <key> <increment> [<timeout>]`

lookup

Description	Performs lookup on the table or subtable for the specified key, and returns its value.
Options	-notouch—When specified, the entry timestamp must not be updated. -subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key subtable name (optional)
Output	value
Full Syntax	`table lookup [-notouch] [-subtable <name>] [--] <key>`

replace

Description	Replaces the value of the specified entry, if it exists.
Options	-notouch—When specified, the entry timestamp must not be updated. -subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key value timeout (default value is 600 seconds, use "indefinite" for entries without expiration) subtable name (optional)
Full Syntax	`table replace [-notouch] [-subtable <name>] [--] <key> <value> [<timeout>]`

set

Description	Creates or updates the specified entry.
Options	-notouch—When specified, the entry timestamp must not be updated. -mustexist—When specified, no increment is performed and the command returns a null string if the table entry does not exist. Mutually exclusive with -excl. -excl—When specified, if the table entry already exists, the entry is not updated and the command returns null string. Mutually exclusive with –mustexist. -subtable—When specified, allows an additional key that groups entries of a certain type.

Input	key value timeout (default value is 600 seconds, use “indefinite” for entries without expiration) subtable name (optional)
Full Syntax	`table set [-notouch] [-subtable <name>] [-mustexist|-excl] [--] <key> <value> [<timeout>]`

timeout

Description	Sets the timeout of the specified entry.
Options	-notouch—When specified, the entry timestamp must not be updated. -mustexist—When specified, no increment is performed and the command returns a null string if the table entry does not exist. Mutually exclusive with -excl. -excl—When specified, if the table entry already exists, the entry is not updated and the command returns null string. Mutually exclusive with –mustexist. -subtable—When specified, allows an additional key that groups entries of a certain type.
Input	key timeout (default value is 600 seconds, use “indefinite” for entries without expiration) subtable name (optional)
Full Syntax	`table timeout [-notouch] [-subtable <name>] [--] <key> <timeout>`

trace

Description

Allows to enable or disable logging, or change log level for a specific session.

Input

Log level

Full Syntax

```
trace
```

Valid Events

All

Implementation Notes

This command is specifically for debug purposes. When using the trace command together with logs on session (/maint/applog/logonses), you can get the trace logs from earlier in the session life, that is, before the trace command was called.

First Implemented Version

33.5.2.0, 33.0.6.0

whereis

Description

Retrieves the geo-location of a specific IP address.

Returns a TCL list that includes: Continent, Country Code, State/Region, City, Zip, Latitude, Longitude when no parameter is specified, or just the specified parameter.

For DPS devices, Perform or Secure subscription is required. If there is no valid license, or the location of the IP address is unknown, the command returns an empty list/parameter.

Traffic is dropped when the IP address is invalid.

Full Syntax

```
whereis <ip address> [continent | country_code | state | city | zip |
latitude | longitude]
```

Valid Events

All events

Implementation Notes

The IP address can also be variable.

Examples:

```
when HTTP_REQUEST {
 set result_list [whereis 15.6.4.9]
 log info $result_list
}

when HTTP_REQUEST {
 set result_list [whereis 15.6.4.9 country_code]
 log info $result_list
}

when HTTP_REQUEST {
 set ip_var "15.6.4.9"
 set result_list [whereis $ip_var latitude]
 log info $result_list
}
```

First Implemented Version

30.5.11.0

Compress Commands

This section describes the following compression commands:

- conf::spath
- conf::service

compress::disable

Description

Disables response compression for the current HTTP transaction.

Full Syntax

```
COMPRESS::disable
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_FILTER_MATCH

HTTP_CRULE_MATCH

HTTP_RESPONSE

Implementation Notes

This command causes an error when there is no compression policy configured for the virtual service or filter.

Decompression occurs if body inspection or modification is required by configured features.

First Implemented Version

30.2.0

compress::enable

Description

Enables response compression for the current HTTP transaction.

Compression parameters are defined via a compression policy attached to the virtual service or filter.

This command is relevant only if compression is already disabled with the conf::spath command.

Full Syntax

```
COMPRESS::enable
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_FILTER_MATCH

HTTP_CRULE_MATCH

HTTP_RESPONSE

Implementation Notes

This command causes an error when there is no compression policy configured for the virtual service or filter.

Enabling compression entails enabling all related functionality, including decompression, when required.

First Implemented Version

30.2.0

CONF Commands

This section describes the following CONF commands:

- conf::spath
- conf::service
- conf::saml

conf::spath

Description

Retrieves the value of the specified attribute in the SecurePath policy.

When calling this command from a virtual service, it retrieves the specified attribute of the SecurePath policy used in that virtual service or its matched content rule. In this case, no additional parameters are required.

When calling this command from a sideband connection, it retrieves the specified attribute of the SecurePath policy used in the service (and optionally its content rule) that sent traffic to that sideband connection. You must specify the service and optionally, the content rule ID (these are internal IDs that are passed on to the sideband connection as metadata and can be retrieved using the SIDEBAND::metadata command).

Valid Events

http_request_events

First Implemented Version

33.5.2.0

Sub-commands

- filesbyps
- apikey
- app_id
- maxreq
- mthdbyps
- qurybyps
- botmng
- name

filesbyps

Description	Retrieves the list of file types to bypass if there are any configured in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String (list of file types to bypass).
Full Syntax	`CONF::spath filebyps [<service id> [<crule id>]]`

apikey

Description	Retrieves the value of the API Key attribute in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String
Full Syntax	`CONF::spath apikey [<service id> [<crule id>]]`

app_id

Description	Retrieves the application id attribute in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String
Full Syntax	`CONF::spath app_id [<service id> [<crule id>]]`

maxreq

Description	Retrieves the value of the maximum request size attribute in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	Integer
Full Syntax	`CONF::spath maxreq [<service id> [<crule id>]]`

mthdbyps

Description	Retrieves the list of HTTP methods to be bypassed if there are any configured in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String
Full Syntax	`CONF::spath mthdbyps [<service id> [<crule id>]]`

qurybyps

Description	Retrieves the value of the query bypass parameter (enabled or disabled) in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String
Full Syntax	`CONF::spath qurybyps [<service id> [<crule id>]]`

botmng

Description	Retrieves whether the botmng protection is enabled or disabled in the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.
Output	String
Full Syntax	`CONF::spath botmng [<service id> [<crule id>]]`

name

Description	Retrieves the name of the SecurePath policy.
Input	• service id (o)—Required only when the command is called from a sideband connection. • cruel id (d)—Required only when the command is called from a sideband connection. Can only be used in conjunction with service id.

Output	String
Full Syntax	`CONF::spath name [<service id> [<crule id>]]`

conf::service

Description

Retrieves the value of the specified attribute in the virtual service configuration.

When calling this command from a virtual service, it retrieves the specified attribute of the that virtual service. In this case, no additional parameters are required.

When calling this command from a sideband connection, it retrieves the specified attribute of the virtual service that sent traffic to that sideband connection. You must specify the service ID (this is an internal ID that is passed on to the sideband connection as metadata and can be retrieved using the SIDEBAND::metadata command).

Valid Events

http_request_events

First Implemented Version

33.5.2.0

Sub-commands

- port
- is_https
- has_crule
- ip_header
- has_spath
- cdn
- parseall

port

Description	Retrieves the virtual service port.
Input	service id (o)—Required only when the command is called from a sideband connection.
Output	Integer
Full Syntax	`CONF::service port [<service id>]]`

is_https

Description	Returns whether the virtual service is HTTPS (has SSL policy attached).
Input	service id (o)—Required only when the command is called from a sideband connection.

Output	Boolean
Full Syntax	`CONF::service is_https [<service id>]`

has_crule

Description	Retrieves whether content rules are configured for the virtual service and of what type.
Input	service id (o)—Required only when the command is called from the sideband connection.
Output	none, http, or ssl
Full Syntax	`CONF::service has_crule [<service id>]`

ip_header

Description	Retrieves the HTTP header that carries the client IP. This is relevant only for CDN/ Proxy deployment.
Input	service id (o)—Required only when the command is called from a sideband connection.
Output	String
Full Syntax	`CONF::service ip_header [<service id>]`

has_spath

Description	Returns whether a SecurePath policy is attached to the virtual service or when called from CONTENT_RULE_MATCH event to the content rule.
Input	service id (o)—Required only when the command is called from a sideband connection.
Output	Boolean
Full Syntax	`CONF::service has_path [<service id>]`

cdn

Description	Returns whether CDN/Proxy deployment is enabled for the virtual service.
Input	service id (o)—Required only when the command is called from a sideband connection.
Output	Boolean
Full Syntax	`CONF::service cdn [<service id>]`

parseall

Description	Returns whether parseall is enabled for the virtual service/filter.
Output	Boolean
Full Syntax	`CONF::service parseall`

conf::saml

Description

Retrieves the value of the specified attribute in the SAML profile.

When calling this command from a virtual service, it retrieves the specified attribute of the SAML profile used in that virtual service or its matched content rule. In this case no additional parameters are required.

Valid Events

http_request_events

Sub-commands

- bind
- consumer
- get_sig_alg
- idpurl
- idpslo
- idpslobind
- should_sign
- spslo

bind

Description	Retrieves the request binding method (the transport mechanism used for the SAML authentication request messages) configured in the SAML profile.
Input	profile_id
Output	1 for redirect, 0 for post
Full Syntax	CONF::saml bind <profile_id>

consumer

Description	Retrieves the value of the response endpoint (relative path).
Input	profile_id

Output	string
Full Syntax	CONF::saml consumer <profile_id>

get_sig_alg

Description	Retrieves the signature algorithm attribute in the SAML profile.
Input	profile_id
Output	string
Full Syntax	CONF::saml get_sig_alg <profile_id>

idpurl

Description	Retrieves the request redirect URL (the IdP URL to which users are redirected for authentication) attribute in the SAML profile.
Input	profile_id
Output	string
Full Syntax	CONF::saml idpurl <profile id>

idpslo

Description	Retrieves the single logout IdP URL (to which logout request will be sent) attribute in the SAML profile.
Input	profile_id
Output	string
Full Syntax	CONF::saml idpslo [profile id>

idpslobind

Description	Retrieves the logout binding method (the transport mechanism used for the SAML logout messages) attribute in the SAML profile.
Input	profile_id
Output	string
Full Syntax	CONF::saml idpslobind <profile id>

should_sign

Description	Retrieves the signature requested attribute in the SAML profile.
Input	profile_id

Output	1/0
Full Syntax	CONF::saml should_sign <profile id>

spslo

Description	Retrieves the single logout SP endpoint (relative path) attribute in the SAML profile.
Input	profile_id
Output	string
Full Syntax	CONF::saml spslo <profile id>

DNS Commands

This section describes the following DNS commands:

DNS::additional

Description

Retrieves, inserts, removes, or clears resource records from the "Additional" section.

Valid Events

DNS_REQUEST
DNS_RESPONSE

First Implemented Version

30.2.0

Sub-commands

- additional
- insert
- remove
- clear

additional

Description	Retrieves all the resource records in the "Additional" section.
Output	TCL list
Full Syntax	`DNS::additional`

insert

Description	Inserts a resource record in the "Additional" section.
Input	Resource record (string).
Full Syntax	`DNS::additional insert <rr>`

remove

Description	Removes a specified resource record from the "Additional" section.
Input	Resource record (string).
Full Syntax	`DNS::additional remove <rr>`

clear

Description	Clears all resource records from the "Additional" section.
Full Syntax	`DNS::additional clear`

DNS::answer

Description

Retrieves, inserts, removes, or clears resource records from the "Answer" section.

Valid Events

DNS_REQUEST
DNS_RESPONSE

First Implemented Version

30.2.0

Sub-commands

- answer
- insert
- remove
- clear

answer

Description	Retrieves all the resource records in the "Answer" section.
Output	TCL list
Full Syntax	`DNS::answer`

insert

Description	Inserts a resource record in the "Answer" section.
Input	Resource record (string).
Full Syntax	`DNS::answer insert <rr>`

remove

Description	Removes a specified resource record from the "Answer" section.
Input	Resource record (string).
Full Syntax	`DNS::answer remove <rr>`

clear

Description	Clears all resource records from the "Answer" section.
Full Syntax	`DNS::answer clear`

DNS::authority

Description

Retrieves, inserts, removes, or clears resource records from the "Authority" section.

Valid Events

DNS_REQUEST

DNS_RESPONSE

First Implemented Version

30.2.0

Sub-commands

- authority
- insert
- remove
- clear

authority

Description	Retrieves all the resource records in the "Authority" section.
Output	TCL list
Full Syntax	`DNS::authority`

insert

Description	Inserts a resource record in the "Authority" section.
Input	Resource record (string).
Full Syntax	`DNS::authority insert <rr>`

remove

Description	Removes a specified resource record from the "Authority" section.
Input	Resource record (string).
Full Syntax	`DNS::authority remove <rr>`

clear

Description	Clears all resource records from the "Authority" section.
Full Syntax	`DNS::authority clear`

DNS::construct_query

Description

Generates a DNS query as a binary string.

Options

- -tcp: The generated query is suitable for DNS over TCP (the first two bytes contain the message size); default is UDP.
- -id: By default, a 16-bit identifier is automatically generated and assigned to each query. This identifier is copied to the corresponding reply and can be used by the requester to match up replies to outstanding queries. This option allows you the specify an identifier, instead of using random generation.
- -inverse: Specifies whether to generate a reverse lookup.
- -rd: Sets Recursion Desired (RD) bit in the generated query. If RD is set, it directs the name server to search for the query recursively.

Input

Name to resolve

Query type

Output

A binary string

Full Syntax

```
DNS::construct_query -name <name> -type <type> [-tcp] [-id <ID>] [-inverse]
[-rd]
```

Valid Events

Any event

First Implemented Version

33.0.1.50

Implementation Notes

Example:

```
set myq [DNS::construct_query -name www.sun.com -type A]
```

DNS::edns0

Description

Gets or sets the values of the extension (EDNS0) presudo resource record.

Valid Events

Any event

First Implemented Version

30.5.0

Sub-commands

- exists
- edns0
- subnet
- del_option
- add_option
- get_option
- has_option
- newrr
- delrr

exists

Description	Checks whether the EDNS0 pseudo-RR exists, or • if the NSID option exists in EDNS0, or • if the client-subnet option exists in EDNS0
Output	true or false
Full Syntax	`DNS::edns0 exists [<nsid|subnet>]`

edns0

Description	Retrieves or changes the value of the specified flag/field.
Input	do—Boolean true/false or an integer (0 <= do is false, other = true) sz—An integer between 0 and 65535 nsid—A byte array value (optional)
Output	do—Returns boolean true/false for the presence of the DO flag. sz—Returns an integer for the UDP size. nsid—Returns a byte array containing the name-server ID or empty if it is an nsid query. A TCL_ERROR is returned if the command is called when there is no EDNS0 or NSID field.
Full Syntax	`DNS::edns0 <do|sz|nsid> [<value>]`

subnet

Description	Retrieves or changes client subnet parameters.
Input	IP (IPv4 or IPv6) source or scope (an integer between 0 and 128) value (optional)
Output	Value of the specified parameter, IP (IPv4 or IPv6) Source or scope (an integer between 0 and 128) Without an argument, returns an IP address, integer, and integer, respectively for IP address, source, and scope. A TCL_ERROR is returned if the command is called when there is no EDNS0 or client-subnet field.
Full Syntax	`DNS::edns0 subnet <address|source|scope> [<IP|int|int>]`

del_option

Description	Deletes one of the ends0 options.
Input	ID of the edns0 option
Output	None (updates current DNS message)
Full Syntax	`DNS::edns0 del_option <option_id>`
Notes	Example: `DNS::edns0 del_option 60006` • If the ends0 OPT pseudo-RR is absent, the command fails. • If the specified option is absent, the command does not perform anything but does not fail. Available since 33.0.1.50.

add_option

Description	Adds an option to edns0 pseudo-RR.
Input	ID of the edns0 option and its value
Output	None (updates current DNS message)
Full Syntax	`DNS::edns0 add_option <id> <value>`
Notes	The value size should be small enough to fit into a DNS query. Available since 33.0.1.50.

get_option

Description	Retrieves the value of the specified edns0 option.
Input	ID of the option

Output	Value
Full Syntax	`DNS::edns0 get_option <id>`
Notes	The command fails if the option is absent or if the edns0 pseudo-RR is absent. Available since 33.0.1.50.

has_option

Description	Checks the presence of the specified edns0 option.
Input	ID of the option
Output	Boolean
Full Syntax	`DNS::edns0 has_option <id>`
Notes	Available since 33.0.1.50.

delrr

Description	Removes the entire edns0 pseudo-RR.
Input	None
Output	None (updates current DNS message)
Full Syntax	`DNS::edns0 delrr`
Notes	The command fails if RR is absent. Available since 33.0.1.50.

newrr

Description	Creates an edns0 pseudo-RR
Input	None
Output	None (updates current DNS message)
Full Syntax	`DNS::edns0 newrr`
Notes	The command fails if edns0 pseudo-RR already exists. The size of RR is hard-coded to 512 bytes (fixed size). Available since 33.0.1.50.

DNS::header

Description

Retrieves or changes various DNS header fields.

Input

One of the following: id|qr|opcode|aa|tc|rd|ra|ad|cd|rcode|qdcount|ancount|nscount|arcount.
Value integer (optional)

Output

For rcode and opcode—string, otherwise integer.

Full Syntax

```
DNS::header <field> [<value>]
```

Valid Events

DNS_REQUEST
DNS_RESPONSE

First Implemented Version

30.2.0

DNS::len

Description

Retrieves the length of the DNS message.

Full Syntax

```
DNS::len
```

Valid Events

DNS_REQUEST
DNS_RESPONSE

First Implemented Version

30.2.0

DNS::message

Description

Returns the content of the current DNS message. This message could have been received by Alteon on a DNS service or the result of the DNS::parse_message command.

Input
A binary string

Output
DNS query as a binary string

Full Syntax
```
DNS::message
```

Valid Events
Any event

First Implemented Version
33.0.1.50

DNS::name

Description
Retrieves or changes the resource record name field.

Input
String (resource record)
Value string (optional)

Output
String

Full Syntax
```
DNS::name <rr> [<value>]
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

DNS::parse_message

Description
Parses the input binary data as a DNS message (query or response) into an internal buffer. This then allows to apply DNS commands that retrieve or change data such as DNS:rr, DNS::edns0, etc.

Optional flag is `[-tcp]`. If the message is received via DNS over TCP, it has 2 bytes of prefix (msg size) before the query itself. Therefore, the parser must know whether or not to strip these 2 bytes.

Input
A binary string

Output
Return value is the size of the message (number of bytes).

Full Syntax
```
DNS::parse_message [-tcp] <data>
```

Valid Events
Any event

Implementation Notes
Updates the internal buffer.

First Implemented Version
33.0.1.50

DNS::ptype

Description
Retrieves the DNS packet type.

Output
One of the following strings:
QUESTION
REFERRAL
ANSWER
NXDOMAIN
NODATA
UNKNOWN

Full Syntax
```
DNS::ptype
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.5.0

DNS::question

Description
Retrieves or changes DNS question fields.

Input
Name, type, or class
Value string (optional)

Output
String

Full Syntax
```
DNS::question <name|type|class> [<value>]
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

DNS::rdata

Description
Retrieves or changes the resource record rdata field.

Input
String (resource record)
Value string (optional)

Output
String

Full Syntax
```
DNS::rdata <rr> [<value>]
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

DNS::release_message

Description
Releases the memory allocated for the DNS message before the session ends to reduce the memory used.

Full Syntax
```
DNS::rdata <rr> [<value>]
```

Valid Events
Any event

First Implemented Version
33.0.2.0

DNS::return

Description
Returns a message to the remote side (client or server) from which the request or response was received.

Full Syntax
```
DNS::return
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

DNS::rr

Description
Builds a resource record. Can specify each parameter separately, or all parameters as a string.

Input
name, type, class, rdata (string)
ttl (integer)

Output
String

Full Syntax
```
DNS::rr <<name> <type> <class> <ttl> <rdata>> | <string>
```

Valid Events

DNS_REQUEST

DNS_RESPONSE

First Implemented Version

30.2.0

DNS::search

Description

Retrieves all occurrences of the specified fields in the specified section. One or more fields can be specified.

Input

name, name_len, type, class, ttl, rdata, rdata_len

Output

A TCL list with all occurrences of the specified field. If multiple fields are specified, a TCL list of lists is returned.

Full Syntax

```
DNS::search <answer|authority|additional|all> [<name>] [<name_len>] [<type>]
[<class>] [<ttl>] [<rdata>] [<rdata_len>]
```

Valid Events

DNS_REQUEST

DNS_RESPONSE

First Implemented Version

30.2.0

DNS::type

Description

Retrieves or changes the resource record type field.

Input

String (resource record)

Value string (optional)

Output

String

Full Syntax

```
DNS::type <rr> [<value>]
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

DNS::ttl

Description
Retrieves or changes the resource record ttl field.

Input
String (resource record)
Value string (optional)

Output
String

Full Syntax
```
DNS::ttl <rr> [<value>]
```

Valid Events
DNS_REQUEST
DNS_RESPONSE

First Implemented Version
30.2.0

HTTP Commands

HTTP commands can access the header and body of an HTTP message, and retrieve or change them. This section describes the following HTTP commands:

HTTP::bypass_proxy

Description

Lets Alteon bypass a proxy server and forward the HTTP request to the required destination. The command performs the following operations:

- Performs DNS resolution for the hostname in the HTTP proxy request URI.
- Transforms the HTTP proxy request into a regular HTTP request by removing the hostname from the URL, and replacing the Proxy-Connection header with a Connection header
- Forwards the HTTP request to the resolved IP address.

Full Syntax

```
HTTP::bypass_proxy
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_FILTER_MATCH

First Implemented Version

29.3.0

HTTP::crule

Description

Accesses a content-based rule, and lets you select, enable or disable that rule, and retrieve the currently matched rule.

Alias

HTTP::class (same sub-commands)

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

First Implementation Version

29.0.0

Sub-commands

- crule
- select

crule

Description	Retrieves the ID of the currently matched content-based rule.
Output	The content-based rule ID.
Full Syntax	`HTTP::crule`

select

Description	Selects a specified content-based rule.
Input	The content-based rule ID.
Full Syntax	`HTTP::crule select <rule ID>`

HTTP::close

Description

Terminates the HTTP connection by inserting a Connection:close header in the HTTP response to the client.

Full Syntax

```
HTTP::close
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_CACHE_RESPONSE
HTTP_RESPONSE_CONTINUE
HTTP_RESPONSE_DATA
HTTP_FILTER_MATCH

Implementation Notes

When called for a request event, this command terminates the HTTP connection when the next response from the server arrives.

First Implemented Version

29.0.0

HTTP::collect

Description

Collects the specified amount of HTTP body data and triggers an HTTP data event (HTTP_REQUEST_DATA or HTTP_RESPONSE_DATA, depending on the context).

Output

The output can be viewed and manipulated using the HTTP::payload command.

Full Syntax

```
HTTP::collect [<length>]
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_RESPONSE
HTTP_CACHE_RESPONSE

HTTP_RESPONSE_DATA

Implementation Notes

- If a length value is not specified, the whole message is collected.
- If a length value is specified, the requested amount at least is collected.
- The HTTP::payload command returns the whole collected message, which may be larger than the specified length.
- If the length of the HTTP message is smaller than the specified length value, the event is still triggered, and the HTTP::payload command returns a buffer smaller than the length value.
- The value of the length parameter is counted from beginning of the payload.
- If subsequent HTTP::collect commands request the same data, no additional DATA event is triggered.

Note: The TCP::collect command behaves differently to the HTTP::collect command.

First Implemented Version

29.0.0

HTTP::content_length

Description

Retrieves the value of the Content-length header (size of the message body in bytes). Valid for both requests and responses.

Output

Integer

Valid Events

all_http_events

First Implemented Version

33.0.2.0

Sub-commands

- content_length <length>

content_length <length>

Description	Modify the content length's value (size of message in bytes). Valid for both requests and responses.
Input	<length>
Output	None

Valid Events	all_http_events
First Implementation Version	33.5.2.0
Implemenation Notes	In case the content-length header does not exist, you receive an error. This command is used to modify the size of the message body in bytes after the headers are already parsed and therefore, the content-length value has already been set. In that case, modifing the header itself (using HTTP::header) does not modify the message body size's value.

HTTP::cookie

Description

Retrieves or changes HTTP cookies. Valid for both requests and responses.

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP_RESPONSE

HTTP_CACHE_RESPONSE

HTTP_RESPONSE_CONTINUE

HTTP_RESPONSE_DATA

Implementation Notes

- When a get or set attribute value command is performed for an irrelevant message type (for example, to retrieve or change the cookie domain on an HTTP request), no action is taken and a null string is returned.
- When a get attribute value command is performed on a valid message type, but such an attribute is not present, a null string is returned.
- When a change attribute value command is performed, on a valid message type, but such attribute is not present, the attribute is added.

Sub-commands

- count
- names
- value
- domain
- path
- version
- ports
- expires

- maxage
- comment
- commenturl
- secure
- samesite
- httponly
- exists
- insert
- remove
- set

count

Description	Retrieves the number of cookies present in the message HTTP headers.
Output	The number of cookies (integer).
Full Syntax	`HTTP::cookie count`

names

Description	Retrieves the names of all the cookies present in the message HTTP headers, in Tcl list format.
Output	The names of all the cookie headers found in the message, in Tcl list format.
Full Syntax	`HTTP::cookie names`

value

Description	Retrieves or changes a specified cookie value.
Input	cookie name cookie value (optional)
Output	The value found in the message for the specified cookie, if it exists; null if the cookie is not present in the message. Relevant only when retrieving a cookie value.
Full Syntax	`HTTP::cookie value <name> [<value>]`
Alias	HTTP::cookie <name> [<value>]

domain

Description	Retrieves or changes a specified cookie domain.
Input	cookie name cookie domain (optional)

Output	The domain of the specified cookie, if it exists; null if the cookie is not present in the message. Relevant only when retrieving a cookie domain.
Full Syntax	`HTTP::cookie domain <name> [<domain>]`

path

Description	Retrieves or changes a specified cookie path.
Input	cookie name cookie path (optional)
Output	The path of the specified cookie, if it exists; null if the cookie is not present in the message. Relevant only when retrieving a cookie path.
Full Syntax	`HTTP::cookie path <name> [<path>]`

version

Description	Retrieves or changes a specified cookie version.
Input	cookie name 0 | 1 | 2 (optional)
Output	The version of the specified cookie, if it exists; null if attribute or cookie are not present in the message. Relevant only when retrieving cookie version.
Full Syntax	`HTTP::cookie version <name> [0 | 1 | 2]`

ports

Description	Retrieves or changes a specified cookie port list for version 2 cookies.
Input	cookie name cookie port list (optional)
Output	The port list of the specified cookie, if it exists; null if the cookie is not present in the message. Relevant only when retrieving a cookie port list.
Full Syntax	`HTTP::cookie ports <name> [<port-list>]`

expires

Description	Retrieves or changes a specified cookie expires attribute. The expiration time provided must be in number of seconds relative to the current time (relative option) or relative to POSIX time (absolute option). Relevant only for version 0 cookies in HTTP responses.
Input	cookie name cookie expiration time in seconds (optional) absolute or relative (only if expiration time is specified). Default is relative.
Output	The expires attribute of the specified cookie, if it exists; null if the cookie is not present in the message.
Full Syntax	`HTTP::cookie expires <name> [<expires>] [absolute | relative]`

maxage

Description	Retrieves or changes a specified cookie max-age attribute. Relevant only for version 1 and 2 cookies in HTTP responses.
Input	cookie name max-age in seconds (optional)
Output	The max-age attribute of the specified cookie, if it exists; null if the attribute or cookie are not present in the message.
Full Syntax	`HTTP::cookie maxage <name> [<max-age>]`

comment

Description	Retrieves or changes a specified cookie comment attribute. Relevant only for version 1 and 2 cookies in HTTP responses.
Input	cookie name comment (optional)
Output	The comment attribute of the specified cookie, if it exists; null if the cookie is not present in the message.
Full Syntax	`HTTP::cookie comment <name> [<comment>]`

commenturl

Description	Retrieves or changes a specified cookie comment URL attribute. Relevant only for version 1 and 2 cookies in HTTP responses.
Input	cookie name comment URL (optional)

Output	The comment URL attribute of the specified cookie, if it exists; null if the cookie is not present in the message.
Full Syntax	`HTTP::cookie commenturl <name> [<comment URL>]`

secure

Description	Retrieves or changes a specified cookie secure attribute. Relevant only for HTTP responses.
Input	cookie name enable or disable (optional)
Output	The secure attribute of the specified cookie, if it exists; null if the attribute or cookie are not present in the message.
Full Syntax	`HTTP::cookie secure <name> [enable | disable]`

samesite

Description	Retrieves or changes specified cookie samesite attribute. Relevant only for HTTP responses.
Input	cookie name lax, strict, or none (optional)
Output	The samesite attribute of the specified cookie, if it exists; null if cookie is not present in the message.
Full Syntax	`HTTP::cookie samesite <name> [lax | strict | none]`

httponly

Description	Retrieves or changes a specified cookie httponly attribute. Relevant only for version 1 and 2 cookies in HTTP responses.
Input	cookie name enable or disable (optional)
Output	The httponly attribute of the specified cookie, if it exists; null if the attribute or cookie are not present in the message.
Full Syntax	`HTTP::cookie httponly <name> [enable | disable]`

exists

Description	Checks if the specified cookie exists in the message headers.
Input	cookie name

Output	True or false
Full Syntax	`HTTP::cookie exists <name>`

insert

Description	Inserts a Cookie header (in an HTTP request) or a Set-Cookie header (in an HTTP response). If such headers already exist, adds the new cookie to the header.
Input	cookie name cookie value cookie domain (optional) cookie path (optional) 0 or 1 or 2 for version (optional) expires/max-age (depending on cookie version) (optional) enable or disable for secure (optional) lax, strict, or none for samesite (optional)
Full Syntax	`HTTP::cookie insert name <name> value <value> [path <path>] [domain <domain>] [version <0 | 1 | 2>] [expires <expires>] [secure <enable | disable>] [samesite <lax | strict | none]` Also supports the short format HTTP::cookie insert <name> <value> for simple cookie insert.
Notes	• If the cookie version is 0, inserts an Expires header when an expires parameter is provided. • If the cookie version is 1 or 2, inserts a Max-Age header. • The order of the parameters in the long format is fixed, but some of the parameters are optional.

remove

Description	Removes a specified cookie.
Options	-all (optional)
Input	cookie name
Full Syntax	`HTTP::cookie remove [-all] <name>`
Notes	• By default, removes only the first occurrence of the cookie name; -all removes all occurrences of the cookie name. • On request, no action is taken, and a null string is returned. Removes the cookie name/value pair. If this is the last name/value pair in the Cookie header, removes the entire header. • On reply, removes the entire Set-Cookie header that matches.

set

Description	Changes cookie attributes. Recommended when multiple attributes must be changed.
Input	cookie name cookie value cookie domain (optional) cookie path (optional) 0 or 1 or 2 for version (optional) expires/max-age (depending on cookie version) (optional) enable or disable for secure (optional) lax, strict, or none for samesite (optional) port list (optional) enable or disable for http only (optional) comment (optional) commenturl (optional)
Full Syntax	`HTTP::cookie set <name> [value <value>] [path <path>] [domain <domain>] [version <0 | 1 | 2>] [expires <expires>] [secure <enable | disable>] [samesite <lax | strict | none>] [ports <port-list>] [httponly <enable | disable>] [comment <comment>] [commenturl <commenturl>]`
Notes	• Certain attributes are only valid in HTTP responses. For example, expires, maxage, secure, samesite, httponly, comment, and commenturl. • Certain attributes are only valid for specific cookie versions. For example, maxage, secure, samesite, httponly, comment, and commenturl for version 1 and 2; expires for version 0; and ports for version 2. • If the cookie version is 0, changes an Expires header when an expires parameter is provided; if the cookie version is 1 or 2, changes the Max-Age header.

HTTP::cookies

Description
Retrieves all HTTP cookies values. Valid for both requests and responses.

Output
String

Full Syntax
```
HTTP:cokies
```

Valid Events
all_http_events

First Implemented Version
33.5.2.0

HTTP::disable

Description
Disables HTTP processing for non-HTTP traffic.

Full Syntax
```
HTTP::disable
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_CONTINUE
HTTP_RESPONSE_DATA
HTTP_FILTER_MATCH

First Implemented Version
29.5.0

HTTP::extension

Description
Retrieves the extension in the Request-URI field.

Output

The extension in the Request-URI field, or a null string if no extension is present.

Full Syntax

```
HTTP::extension
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP::header

Description

Retrieves or changes HTTP headers in load balanced HTTP messages. Valid for both request and responses. Not valid for HTTP messages generated by Alteon.

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP_RESPONSE

HTTP_CACHE_RESPONSE

HTTP_RESPONSE_CONTINUE

HTTP_RESPONSE_DATA

First Implemented Version

29.0.0

Implementation Notes

- Only informational sub-commands are allowed in the HTTP_RESPONSE_CONTINUE event.
- It is the responsibility of the user to provide a valid header name and value. For example, if the header name provided to the "insert" sub-command contains whitespace, the command succeeds and a packet with an invalid header enters the network.

Sub-commands

- count
- names
- value
- values
- exists
- at

- insert
- replace
- remove
- is_redirect
- is_keepalive

count

Description	Retrieves the number of headers present in the HTTP message, or optionally the number of occurrences of a certain header. Counts multiple occurrences of the same header name separately.
Input	The header name (optional).
Output	An integer (the number of headers).
Full Syntax	`HTTP::header count [<name>]`

names

Description	Retrieves the names of all the headers present in the HTTP message, in Tcl list format. Retrieves multiple occurrences of the same header name multiple times.
Output	The names of all the headers found in the message, in Tcl list format.
Full Syntax	`HTTP::header names`

value

Description	Retrieves a specified header value. If the same header name appears multiple times in the message, retrieves only the value of the last occurrence.
Input	The header name.
Output	The value found in the message for the specified header, or an empty string if the header is not present in the message.
Full Syntax	`HTTP::header value <name>`
Alias	HTTP::header <name>

values

Description	Retrieves the values of all occurrences of the specified header.
Input	The header name.
Output	All values found in the message for the specified header, in Tcl list format, or an empty string if the header is not present in the message.
Full Syntax	`HTTP::header values <name>`
Notes	Retrieves last header value only.

exists

Description	Checks if the specified header is present in the message.
Input	The header name.
Output	1 (for true) and 0 (for false)
Full Syntax	`HTTP::header exists <name>`

at

Description	Retrieves the name of the i-th header in the message.
Input	The header index.
Output	The header name.
Full Syntax	`HTTP::header at <index>`

insert

Description	Inserts a header or headers in the HTTP headers area at the end of the HTTP message.
Input	Header name-value pairs.
Full Syntax	`HTTP::header insert [<name> <value>]+`

replace

Description	Replaces the value of the last occurrence of the specified header, if it exists; else inserts the specified header.
Input	header name value
Full Syntax	`HTTP::header replace <name> <value>`

remove

Description	Removes the specified header. If the header appears multiple times, removes only the last occurrence.
Input	The header name.
Full Syntax	`HTTP::header remove <name>`

is_redirect

Description	Checks if the Status Code on the HTTP response is a redirect code (301, 302, 303, 305 or 307).
Input	True or false.
Full Syntax	`HTTP::header is_redirect`
Notes	Valid only for responses.

is_keepalive

Description	Checks if keep-alive is enabled on the current connection.
Input	True or false.
Full Syntax	`HTTP::header is_keepalive`

HTTP::headers

Description

Removes or replaces the entire HTTP headers section in a message (not valid for HTTP messages generated by a device). Valid for both requests and responses.

Valid Events

All HTTP events.

Implementation Notes

- Only informational sub-commands are allowed in the HTTP_RESPONSE_CONTINUE event.
- It is the responsibility of the user to provide a valid header name and value. For example, if the header name provided to the "insert" sub-command contains whitespace, the command succeeds and a packet with an invalid header enters the network.

First Implemented Version

33.0.1.50

Sub-commands

- remove
- replace

remove

Description	Removes all the headers in the HTTP message including trailing \r\n\r\n.
Input	None
Output	None

Full Syntax	`HTTP::headers remove`
Notes	Available since 33.0.1.50.

replace

Description	Replaces the entire headers section in the HTTP message including trailing \r\n\r\n with new data supplied by the user.
Input	A binary or text string
Output	None
Full Syntax	`HTTP::headers replace <data>`
Notes	Available since 33.0.1.50.

HTTP::host

Description
Retrieves the value of the Host header. Relevant for HTTP requests only.

Output
The value of the Host header.

Full Syntax
```
HTTP::host
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

First Implemented Version
29.0.0

HTTP::method

Description
Retrieves the method from the HTTP request.

Output
The HTTP method.

Full Syntax
```
HTTP::header method
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

HTTP::path

Description
Retrieves or changes the path in the Request-URI field. The query string is not included.

Input

The path (optional).

Output
The path in the Request-URI field.

Full Syntax
```
HTTP::path [path]
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

Implementation Notes
When the change path command is applied, but there is no path, the specified path is inserted. For example, HTTP::path (index.html).

HTTP::param

Description
Extract the value of the specified parameter name from a list of parameters separated by delimiters.

Input
param key - the name of the parameter whose value needs to be extracted.
param string - the list of parameters.
delimiters (o) - the possible delimiters between parameters. The default delimiters are "?" and "&".

Output
string

Full Syntax
HTTP::param <param key><parameters string>[{<delimiters>, "?&"}]

Valid Events
all_icap_and_http_events

First Implemented Version
34.5

Example
HTTP::param "b" "a=1;b=2;c=3" ";" will return "2".

HTTP::percent_encode

Description

Performs percent-encoding on the specified string - i.e replacing " " (space) with %20. Percent-encoding, alsoknown as URL encoding, is a method to encode arbitrary data in a uniform resource identifier (URI) using only the US-ASCII characters legal within a URI.

See details at https://en.wikipedia.org/wiki/Percent-encoding.

Input

string

Output

string

Full Syntax

HTTP::percent_encode <content>

Valid Events

http_request_and_lb_events

First Implemented Version

34.5

HTTP::percent_decode

Description

Reverses "percent_encode" on the specified string - i.e. replace %20 with " " (space). Percent-encoding, alsoknown as URL encoding, is a method to encode arbitrary data in a uniform resource identifier (URI) using only the US-ASCII characters legal within a URI.

See details at https://en.wikipedia.org/wiki/Percent-encoding.

Input

string

Output

string

Full Syntax

HTTP::percent_decode <content>

Valid Events

http_request_and_lb_events

First Implemented Version

34.5

HTTP::password

Description

Retrieves the basic authentication password from the Authorization header in the HTTP request, after performing base64 decoding on the basic authentication user and password. Returns an empty string if the Authorization header does not exist.

Output

The authorization password.

Full Syntax

```
HTTP::password
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP::payload

Description

Retrieves or changes the HTTP body (does not include headers). Valid for both request and responses.

Valid Events

HTTP_REQUEST_DATA

HTTP_RESPONSE_DATA

Sub-commands

- payload
- length
- replace
- find
- find_all
- clear

payload

Description	Retrieves the payload collected. Either all of the payload, the specified number of bytes, or the payload from a specified offset position.
Input	offset (optional) length (optional)—mandatory if offset appears

Output	HTTP body
Full Syntax	`HTTP::payload [[<offset>] <length>]`

length

Description	Retrieves the length of the collected payload, in bytes.
Output	payload length (integer)
Full Syntax	`HTTP::payload length`

replace

Description	Replaces all or part of the payload. You can specify where the replacement starts (offset), how much should be replaced (length), and the content of the replacement string.
Full Syntax	`HTTP::payload replace <offset> <length> <new_string>`

find

Description	Searches for the first occurrence of a string in the collected payload and, if found, returns its byte length offset within the payload. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns -1 if no occurrence is found.
Options	-nocase
Input	string start_from_offset (optional)
Output	Offset or -1 (integer)
Full Syntax	`HTTP::payload find [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [HTTP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the HTTP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

find_all

Description	Searches for all non-overlapping occurrences of a string in the collected payload. Returns a list of byte-length offsets. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns an empty list if no occurrence is found.
Options	-nocase

Input	string start_from_offset (optional)
Output	A list of occurrences (Tcl list of integers).
Full Syntax	`HTTP::payload find_all [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [HTTP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the HTTP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

clear

Description	Clears the current payload. This is a shortcut for HTTP::payload 0 [HTTP::payload length] ""
Full Syntax	`HTTP::payload clear`

HTTP::query

Description
Retrieves the query in the Request-URI field.

Output
The query in the Request-URI field, or a null string if no query is present.

Full Syntax
```
HTTP::query
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

HTTP::redirect

Description
Redirects the HTTP request or response to the specified URL.

Input
URL status code (301, 302, 303, 307, or 308). Default is 302. It is the responsibility of the user to make sure that 303 and 307 are used only in HTTP 1.1 messages.

Full Syntax
```
HTTP::redirect <URL> [301 | 302 | 303 | 307 | 308]
```

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_CACHE_RESPONSE
HTTP_RESPONSE_DATA

Implementation Notes
Generates a message to the client with the redirect status code and Location header with the provided URL. In some cases, this command prevents keep-alive on this connection.

HTTP::replace_all

Description
Replaces all HTTP content (headers and body).

Full Syntax
```
HTTP::replace_all <data>
```

Valid Events
all_response_events

First Implemented Version
33.5.2.0

HTTP::request

Description
Retrieves the entire HTTP headers area from the message.

Output
The relevant part of the HTTP request headers string (without the trailing \r\n).

Full Syntax
HTTP::request

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

HTTP::req_count

Description
Reports the number of HTTP requests reaching the current connection.

Output
An integer counter.

Full Syntax
```
HTTP::req_count
```

Alias
HTTP::request_num

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP_RESPONSE

HTTP_CACHE_RESPONSE

HTTP_RESPONSE_CONTINUE

HTTP_RESPONSE_DATA

HTTP::respond

Description

Sends a response message with specified headers and content to the client. When called for a server-side connection, the response replaces the response sent by the server. The Content-Length header is calculated and added automatically.

Input

status code

content (optional)

header name-value pairs (optional)

Full Syntax

```
HTTP::respond <status code> [content <string>] [<header name> <header value>]+
```

Valid Events

HTTP_REQUEST

HTTP_REQUEST_DATA

HTTP_CRULE_MATCH

HTTP_CRULE_NOMATCH

HTTP_RESPONSE

HTTP_CACHE_RESPONSE

HTTP_RESPONSE_DATA

Implementation Notes

The response message generated with this command is not subject to configured HTTP processing (such as compression or URL rewriting). In some cases, this command prevents keep-alive on this connection.

HTTP::retry

Description

Re-sends the specified valid request to the server. A new server can be selected for this new

request.

Input
A valid HTTP request string up to 1k of data. Usually assigned from the HTTP::request command in the previous transaction.

Full Syntax
```
HTTP::respond <request string>
```

Valid Events
HTTP_RESPONSE

HTTP_RESPONSE_DATA

First Implemented Version
30.5.0

Implementation Notes
May fail if the request message is too long, or the previous request is still being processed.

HTTP::status

Description
Retrieves the status code from the current HTTP response.

Output
The status code.

Full Syntax
```
HTTP::status
```

Valid Events
HTTP_RESPONSE

HTTP_CACHE_RESPONSE

HTTP_RESPONSE_CONTINUE

HTTP_RESPONSE_DATA

HTTP::transform_request

Description
Transforms an HTTP request into an HTTP proxy request. Inserts the Host header value, or destination IP address if no Host header is present, in the request URL.

Full Syntax
```
HTTP::transform_request
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_FILTER_MATCH

HTTP::uri

Description

Retrieves or changes the Request-URI field (path and query).

Input

The URI in the Request-URI field (optional).

Output

The Request-URI field.

Full Syntax

```
HTTP::uri [<uri>]
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

HTTP::username

Description

Retrieves the basic authentication user name from the Authorization header in the HTTP request, after performing base64 decode on the basic authentication user and password. Returns an empty string if Authorization header does not exist.

Output

The authorization user name.

Full Syntax

```
HTTP::username
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH

HTTP::version

Description

Retrieves or changes the HTTP version in the current message.

Output

The HTTP version.

Full Syntax

```
HTTP::version [<"0.9" | "1.0" | "1.1">]
```

Valid Events

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_CACHE_RESPONSE
HTTP_RESPONSE_CONTINUE
HTTP_RESPONSE_DATA

IP Commands

IP commands provide read access to IP header fields.

This section describes the following IP commands:

IP::addr

Description

A utility for IP address manipulation that performs the following operations:

- Subnet calculation
- Byte-to-string IP address parsing
- IP address comparison

First Implemented Version

29.3.0

Sub-commands

- addr
- parse

addr

Description	Retrieves the group ID, and the IP address and port of the server selected by the load balancing algorithm. If no server is selected, returns only the group ID.
Input	IP address Mask/prefix (optional)
Output	Subnet
Full Syntax	`IP::addr [-ipv4 | -ipv6] <address> mask <mask/prefix>`

parse

Description	Parses a binary format IP address into string format.
Options	swap
Input	IP value and prefix (optional) for each of the IP addresses.
Output	The IP address in string format.
Full Syntax	`IP::addr parse [-ipv4 [-swap] | -ipv6] <bytearray> [<offset>]`

IP::client_addr

Description

Retrieves the current connection client IP address.

Output

The current connection client IP address.

Full Syntax

```
IP::client_addr
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version

29.0.0

IP::local_addr

Description

Retrieves the Alteon IP address for the connection, depending on the context—virtual server VIP on the client side, and the client IP or PIP address (if the PIP is configured) on the server side.

Output

The Alteon IP address.

Full Syntax

```
IP::local_addr
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version
29.0.0

IP::protocol

Description
Retrieves the upper layer (L4) protocol number.

Output
An integer.

Full Syntax
```
IP::protocol
```

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE

HTTP_CACHE_RESPONSE

First Implemented Version
29.0.0

IP::remote_addr

Description
Retrieves the remote client or server IP address for a connection, depending on the context.

Output
The IP address for the remote client or server.

Full Syntax
```
IP::remote_addr
```

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version
29.0.0

IP::server_addr

Description
Retrieves the server IP address for a connection.

Output
The IP address for the server. Returns 0 if a server-side connection is not yet established.

Full Syntax

```
IP::server_addr
```

Valid Events

SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version

29.0.0

IP::version

Description

Retrieves the IP version used in the present connection (4 or 6).

Output

An integer (4 or 6).

Full Syntax

```
IP::version
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

First Implemented Version
29.0.0

LB Commands

LB commands provide access to load balancing selections.

This section describes the following LB commands:

LB::server

Description
Retrieves data regarding the server selected by the load balancing mechanism.

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE

First Implemented Version
29.0.0

Sub-commands
- server
- group
- addr
- port
- id

server

Description	Retrieves the group ID, and the IP address and port of the server selected by the load balancing algorithm. If no server is selected, returns only the group ID.
Output	Tcl list (group ID, real server IP, server port)
Full Syntax	`LB::server`
Alias	LB::server name

group

Description	Retrieves the group ID selected by the load balancing algorithm.
Output	integer (group ID)
Full Syntax	`LB::server group`
Alias	LB::server pool

addr

Description	Retrieves the IP address of the server selected by the load balancing algorithm. If no server is selected, returns an empty string.
Output	IP address
Full Syntax	`LB::server addr`

port

Description	Retrieves the TCP/UDP port of the server selected by the load balancing algorithm. If no server is selected, returns zero.
Output	integer (port)
Full Syntax	`LB::server port`

id

Description	Retrieves the ID of the real server that was selected by the load balancing algorithm. If no server is selected, returns null.
Output	integer (ID)
Full Syntax	`LB::server id`

LB::status

Description

Retrieves the status of a selected server (if one is selected), or of a specified real server instance (group, server and port must be specified).

Input

Group ID (optional)

Server IP address (optional)

Port (optional)

Output

Status

Full Syntax

```
LB::status [group <group id> server <ID|IP> [<port>]]
```

Valid Events

LB::status without group, server—All server-side traffic events, LB_SELECTED.

LB::status with group, server—All traffic events.

First Implemented Version

30.0.0

LB::reselect

Description

Selects the next server in the group.

Full Syntax

```
LB::reselect [group <ID> [server <ID|IP> [<port>]]]
```

Alias

```
LB::reselect [pool <ID> [member <ID|IP> [<port>]]]
```

Valid Events

LB_SELECTED

LB_FAILED, PERSIST_DOWN

First Implemented Version

30.0.0

RADIUS Commands

This section describes the following RTSP commands:

RADIUS::avp

Description

Retrieves or adds/changes/removes one of the AVPs of the packet.

Valid Events

radius_data_events

First Implemented Version

30.5

Sub-commands

- avp
- delete
- insert
- replace
- clear_all

avp

Description	Retrieves the value of the selected attribute. The value type (octet, string, integer, ip4, ip6) can be specified. By default it is as described in the RADIUS AVPs list document. An index (optional), vendor id, and vendor type (also optional).
Input	Attribute (value 1–255, or optionally a unique name from this list: IANA). The attr type is optional and can be one of the following: octet|string|ip4|ip6|integer. If the type was not selected the default type of the field will be as described in the RADIUS AVPs list document, and the default type for all vendor-specific attributes is octet. The Index value is for attributes and Vendor attributes that have multiple instances (some attributes can appear more than once). The default index is zero if no index was selected. The vendor ID is a number between 0 and (2^32)-1 (4 bytes data). The vendor type is the unique code, a number between 0 and 255.
Output	The value found in the message for the specified attribute; empty string if the attribute is not found or on command failure.

Full Syntax	`RADIUS::avp <attr> [<attr_type>] [index <i>] [vendor-id <ID> vendor-type <type>]`
Notes	Invokes an error message when: 1. Wrong input 2. Wrong command usage 3. Command failure

delete

Description	Removes the specified attribute from the RADIUS message.
Input	Attribute (value 1–255, or optionally a unique name from this list: IANA). The Index value is for attributes and Vendor attributes that have multiple instances (some attributes can appear more than once). The default index is zero if no index was selected. The vendor ID is a number between 0 and (2^32)-1 (4 bytes data). The vendor type is the unique code, a number between 0 and 255.
Full Syntax	`RADIUS::avp delete <attr> [index <i>] [vendor-id <ID> vendor-type <type>]`
Notes	Invokes an error message when: 1. Wrong input 2. Wrong command usage 3. Command failure

insert

Description	Inserts the specified attribute and value into the RADIUS message. The new attribute is inserted as the first AVP of the packet.
Input	Attribute (value 1–255, or optionally a unique name from this list: IANA). The value is the data that will be saved in this attribute. The attr type is optional and can be one of the following: octet|string|ip4|ip6|integer. If the type was not selected the default type of the field will be as described in the RADIUS AVPs list document. The vendor ID is a number between 0 and (2^32)-1 (4 bytes data). The vendor type is the unique code, a number between 0 and 255.
Full Syntax	`RADIUS::avp insert <attr> <value> [<attr_type>] [vendor-id <ID> vendor-type <type>]`
Notes	Invokes an error message when: 1. Wrong input 2. Wrong command usage 3. Command failure

replace

Description	Replaces the value of the specified attribute, if it exists.
Input	Attribute (value 1–255, or optionally a unique name from this list: IANA). The attr type is optional and can be one of the following: octet|string|ip4|ip6|integer. If the type was not selected the default type of the field will be as described in the RADIUS AVPs list document. The Index value is for attributes and Vendor attributes that have multiple instances (some attributes can appear more than once). The default index is zero if no index was selected. The vendor ID is a number between 0 and (2^32)-1 (4 bytes data). The vendor type is the unique code, a number between 0 and 255.
Full Syntax	`RADIUS::avp replace <attr> <value> [<attr_type>] [index <i>] [vendor-id <ID> vendor-type <type>]`
Notes	Invokes an error message when: 1. Wrong input 2. Wrong command usage 3. Command failure

clear_all

Description	Removes all the AVPs from the packet.
Full Syntax	`RADIUS::avp clear_all`
Notes	Invokes an error message when: 1. Wrong input 2. Wrong command usage 3. Command failure

SAML Commands

This section describes the following sideband commands:

- Saml::verify, page 146

Saml::sign

Description

Signs SSO or SLO SAML request, when using Redirect bind (signature appears as seprate query parameter in Redirect mode).

Input

profile id

contents (request string)

Output

binary signature

Full Syntax

SAML::sign <profile id> <contents>

Valid Events

http_request_and_lb_events

First Implemented Version

34.5

Saml::process_slo_response

Description

Validates SLO response and retrieves the InResponseTo and StatusCode parameters from the SLO response .

Input

profile id

slo bind method

saml session key (created during sso and extracted from the client cookie)

slo response

Output

TCL dictionary (name=value container), with two values: is_valid (StatusCode) and the value of in_response_to (InResponseTo) field

Full Syntax

SAML::process_slo_response <profile id> <bind> <session key> <slo response>

Valid Events

http_request_and_lb_events

First Implemented Version
34.5

Saml::process_response

Description
Validates SAML SSO response and retrieves the different attributes from the authentication response.

Input
profile id

sso assertion

Output
TCL dictionary (name=value container), with several values, including the assertion attributes.

Full Syntax
SAML::process_response <profile id> <response>

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::is_endpoint

Description
Checks if the specified relative path is an endpoint for the application, and what type of endpoint.

Input
relative path (string)

Output
endpoint type

- 1 for sso endpoint
- 2 for slo endpoint
- 0 if no endpoint

saml profile name or empty string in case this is not endpoint

Full Syntax
SAML::is_endpoint <path>

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::get_saml_profile

Description
Retrieves the currently active SAML profile for the session, either from the matched content-rule of from the service.

Output
name of saml profile if found, empty string if not found

Full Syntax
SAML::get_saml_profile

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::get_expiration_date

Description
Calculates the SAML session cookie expiration date from the number of seconds remaining until expiration.

Input
expiration seconds

Output
date (string)

Full Syntax
SAML::get_expiration_date <expiration seconds>

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::generate_slo_request

Description
Generates a single logout request.

Input
profile id
request id
subject name id

Output
saml slo request (string)

Full Syntax
SAML::generate_slo_request <profile id> <request id> <subject name id>

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::generate_request

Description
Generates an authentication request (SAML SSO request).

Input
profile id
request id

Output
saml request (string)

Full Syntax
SAML::generate_request <profile id> <request id>

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::generate_id

Description
Generates random request id for a new SAML SSO or SLO request.

Output
string

Full Syntax
SAML::generate_id

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::stats

Description
Update SAML cookie statistics from SAML AS++ script for both cookie valid and cookie invalid cases.

Input
"inc" (increment)
profile id
<"auth_invalid_cookies" | "auth_successful_cookies">

Full Syntax
SAML::stats "inc" <profile id> <"auth_invalid_cookies"|"auth_successful_cookies">

Valid Events
http_request_and_lb_events

First Implemented Version
34.5

Saml::verify

Description

Verifies the signature of an SSO or SLO response, that arrived in a query string argument (relevant to Redirect binding only).

Input

profile id

signing algorithm

contents

signature

Output

Boolean

Full Syntax

SAML::verify <profile id><algorithm><contents><signature>

Valid Events

http_request_and_lb_events

First Implemented Version

34.5

Sideband Commands

Comment–Hidden for 33.0.1 since these commands apply to 33.0.2 doc (implemented in 33.0.1.50).

This section describes the following sideband commands:

Sideband::add_action

Description

Allows to add actions that should be performed in the main session, based on the sideband response.

Valid Events

http_response_events

First Implemented Version

33.5.2.0

Implementation Notes

After all required actions are specified, you must call the SIDEBAND::serialize command to get all the Sideband actions arranged in order and in a binary mode. The resulting binary string should be then used to replace the HTTP request (HTTP::replace_all). The main session gets the actions in binary form, deserializes them, and then applies them.

Sub-commands

js_inject

Description	Adds an action to inject the specified JS script to the HTTP response from the server before forwarding the response to the client.
Input	JS script (string)
Output	None
Full Syntax	SIDEBAND::add_action js_inject <js_payload>

send_response

Description	Adds an action to send the specified response to the client.
Input	HTTP response message (string).
Full Syntax	SIDEBAND::add_action send_response <response_message>

request_header

Description	Adds an action to insert the specified header to the original HTTP request before forwarding it to the server.
Input	Header name and header value (strings)
Output	None
Full Syntax	SIDEBAND::add_action request_header <header_name> <header_value>

response_header

Description	Adds an action to insert the specified header to the server response before forwarding it to the client.
Input	Header name and header value (strings)
Output	None
Full Syntax	SIDEBAND::add_action response_header <header_name> <header_value>

add_cookie

Description	Adds an action to insert the specified cookie to the server response before forwarding it to the client.
Input	Cookie (cookie key and cookie value) Example: SIDEBAND::add_action add_cookie "uzmd=1654003942; HttpOnly;"
Output	None
Full Syntax	SIDEBAND::add_action add_cookie <cookie_payload>

remove_cookie

Description	Adds an action to remove the specified cookie from the original HTTP request before forwarding the response to the server.
Input	• cookie name (string) • prefix or exact—To specify how to match the cookie name.
Output	None
Full Syntax	SIDEBAND::add_action remove_cookie <cookie_name> <prefix/exact>

terminate_session

Description	Adds an action to terminate the session to the client.
Input	None
Output	None
Full Syntax	SIDEBAND::add_action terminate_session

Sideband::metadata

Description

Retrieves the metadata inserted in the Sideband request by the main session.

Input

One of the following:

- client_ip—TCP source IP
- service_id—Internal virtual service ID
- crule_id—Content rule ID

Output

None

Fill Syntax

SIDEBAND::metadata <client_ip|service_id|crule_id>

Valid Events

http_message_events

First Implemented Version

33.5.2.0

Sideband::payload

Description

Retrieves or manipulates TCP payload collected up to this time.

Valid Events

Any event

Sub-commands

- none

none

Description	Retrieves payload.
Input	None
Output	String
Full Syntax	`SIDEBAND::payload`
Notes	Available since 33.0.1.50.

Sideband::respond

Description
Sends a response to the client.

Input
Response data

Output
None

Full Syntax
SIDEBAND::respond <data

Valid Events
sideband_response_events

First Implemented Version
33.0.1.0

Sideband::send

Description
Sends the specified data message through the sideband connection.

Valid Events
HTTP_REQUEST
HTTP_REQUEST_DATA

Sub-commands
- none

none

Description	Sends data through the sideband connection.
Input	Data
Output	None
Full Syntax	`SIDEBAND::send <data>`
Notes	Available since 33.0.1.50.

Sideband::serialize

Description

Requests to serialize the sideband actions. This is necessary to arrange, in proper order and binary format, all the actions that must be performed on the main session. The actions are based on the response received on the sideband connection.

Output

Sideband actions in binary form

Full Syntax

SIDEBAND::serialize

Valid Events

http_response_events

Implementation Notes

If no action was added, the serialize output is empty. The main session interprets this as to allow the original request to be forwarded to the server.

First Implemented Version

33.5.2.0

Sideband::age {ms_time}

Description

This commands allows changing the configured sideband timeout, only for the transaction on which the command runs.

Full Syntax

SIDEBAND::age {ms_time}

Valid Age

10 milliseconds to 300000 milliseconds (5 minutes)

Valid Events

http_request, http_request_data

Implementation Notes

Applicable only for Sideband of type http, only on AS++ that attached to sideband.

First Implemented Version

33.5.11.0

SIP Commands

This section describes the following SIP commands:

SIP::call_id

Description

Retrieves the Call ID value from a SIP message.

Full Syntax

```
SIP::call_id
```

Valid Events

SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

SIP::discard

Description

Discards the SIP message.

Full Syntax

```
SIP::discard
```

Valid Events

SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

SIP::from

Description

Retrieves the From header value from a SIP message.

Full Syntax

```
SIP::from
```

Valid Events

SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

SIP::header

Description

Retrieves or changes SIP headers. Valid for both requests and responses.

First Implemented Version

29.4.0

Sub-commands

- header
- insert
- remove
- replace

header

Description	Retrieves the value of the specified header. If the header appears multiple times, you can specify an index. By default, the command returns the first occurrence.
Input	Header name
Output	The value found in the message for the specified header. An empty string if the header is not present in the message.
Full Syntax	`SIP::header <name> [<index>]`
Alias	`SIP::header value <name> [<index>]`

insert

Description	Inserts the header in the SIP message. If the header already exists, the command inserts the new header before the first existing occurrence. If no such header exists, the command inserts the SIP::via header at the beginning of the SIP headers area, and inserts all other headers at the end of the SIP headers area.
Input	Header name Header value
Full Syntax	`SIP::header insert <name> <value>`

remove

Description	Removes the specified header. If the header appears multiple times, you can specify which index to remove. By default, the command removes the first occurrence.
Input	Header name
Full Syntax	`SIP::header remove <name> [<index>]`

replace

Description	Replaces the value of the specified occurrence of a header. If the header does not exist, the command inserts the header.
Input	Header name Header value
Full Syntax	`SIP::header replace <name> <value> [<index>]`

SIP::method

Description

Retrieves the method from a SIP request.

Full Syntax

```
SIP::method
```

Valid Events

SIP_REQUEST

SIP::payload

Description

Retrieves the SIP message payload.

Full Syntax

```
SIP::payload
```

Valid Events

SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

SIP::respond

Description

Sends a SIP response to the request originator.

Full Syntax

```
SIP::respond <status code> [<reason phrase>] [<header name> <header value>]+
```

Valid Events

SIP_REQUEST

SIP::response

Description

Retrieves or changes the SIP response status code and reason phrase.

First Implemented Version

29.4.0

Sub-commands

- code
- phrase
- rewrite

code

Description	Retrieves the SIP response status code.
Output	status code
Full Syntax	`SIP::response code`

phrase

Description	Retrieves the SIP response reason phrase.
Output	reason phrase (string)
Full Syntax	`SIP::response phrase`

rewrite

Description	Rewrites the SIP response status code and/or reason phrase.
Input	status code reason phrase (optional)
Full Syntax	`SIP::response rewrite <status code> [<reason phrase>]`

SIP::to

Description
Retrieves the To header value from a SIP message.

Full Syntax
```
SIP::to
```

Valid Events
SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

SIP::uri

Description
Retrieves or changes the SIP Request-URI.

Input
uri (optional)

Full Syntax
```
SIP::uri [<uri>]
```

Valid Events
SIP_REQUEST
SIP_REQUEST_SEND

SIP::via

Description
Retrieves the value of the Via SIP header.

Valid Events
SIP_REQUEST
SIP_REQUEST_SEND
SIP_RESPONSE
SIP_RESPONSE_SEND

First Implemented Version
29.4.0

Sub-commands

- via
- proto
- sent_by
- received
- branch
- maddr
- ttl

via

Description	Retrieves the value of the Via header at the specified index. By default, retrieves the first occurrence value.
Input	index (optional)
Output	string
Full Syntax	`SIP::via [<index>]`

proto

Description	Retrieves the protocol parameter from the Via header at the specified index.
Input	index (optional)
Output	string
Full Syntax	`SIP::via proto [<index>]`

sent_by

Description	Retrieves the sent_by parameter from the Via header at the specified index.
Input	index (optional)
Full Syntax	`SIP::via sent_by [<index>]`

received

Description	Retrieves the received parameter from the Via header at the specified index.
Input	index (optional)
Full Syntax	`SIP::via received [<index>]`

branch

Description	Retrieves the branch parameter from the Via header at the specified index.
Input	index (optional)
Full Syntax	`SIP::via branch [<index>]`

maddr

Description	Retrieves the maddr parameter from the Via header at the specified index.
Input	index (optional)
Full Syntax	`SIP::via maddr [<index>]`

ttl

Description	Retrieves the ttl parameter from the Via header at the specified index.
Input	index (optional)
Full Syntax	`SIP::via ttl [<index>]`

SSL Commands

SSL commands can enable or disable SSL on connection.

This section describes the following SSL commands:

SSL::enable

Description
Enables the SSL or TLS protocol in the current client-side and server-side connection.

Full Syntax
```
SSL::enable
```

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
SERVER_CONNECTED
SERVER_DATA

Implementation Notes
This command causes an error when there is no SSL policy configured for the service.

SSL::disable

Description
Disables the SSL or TLS protocol in the current client-side and server-side connection.

Full Syntax
```
SSL::disable
```

Valid Events
CLIENT_ACCEPTED
CLIENT_DATA
SERVER_CONNECTED
SERVER_DATA

Implementation Notes
SSL remains disabled on the server-side connection until a new server connection is established.

SSL::cert

Description
Retrieves an X509 certificate or related data.

Valid Events
CLIENTSSL_CLIENTCERT
CLIENT_DATA
CLIENT_CLOSED

HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
LB_SELECTED
LB_FAILED
PERSIST_DOWN

First Implemented Version

30.0.0

Sub-commands

- cert
- issuer
- count

cert

Description	Retrieves the specified certificate in the chain. For the first certificate the index is 0.
Input	Index
Output	X509 certificate (string)
Full Syntax	`SSL::cert <index>`

issuer

Description	Retrieves the issuer certificate of the specified certificate in the chain.
Input	Index
Output	X509 certificate (string)
Full Syntax	`SSL::cert issuer <index>`

count

Description	Retrieves the number of certificates in the chain.
Output	integer
Full Syntax	`SSL::cert count`

SSL::cipher

Description

Retrieves information regarding the SSL cipher in use.

Valid Events

CLIENTSSL_CLIENTCERT
CLIENT_DATA
CLIENT_CLOSED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
LB_SELECTED
LB_FAILED
PERSIST_DOWN

First Implemented Version

30.0.0

Sub-commands

- name
- version
- bits

name

Description	Retrieves the name of the SSL cipher in use, in OpenSSL format.
Output	String
Full Syntax	`SSL::cipher name`

version

Description	Retrieves the version of the SSL cipher in use.
Output	String
Full Syntax	`SSL::cipher version`

bits

Description	Retrieves the number of bits of the SSL cipher used.
Output	integer
Full Syntax	`SSL::cipher bits`

SSL::cipherlist

Description

Set SSL/TLS ciphers to use in Backend SSL Client Hello.

Full Syntax

SSL::cipherlist <ciper1:cipher2...>

Input

string of ciper-suites, seperated by colon, in OpenSSL format

Valid Events

SERVERSSL_CLIENTHELLO_SEND

First Implemented Version

34.5.4.0

Implementation Notes

Not applicable to to the special images that support Chinese SM ciphers.

SSL::mode

Description

Indicates whether SSL encryption is in use.

Returns 1 if the current context has SSL encryption, otherwise returns 0.

Full Syntax

```
SSL::mode
```

Valid Events

All

First Implemented Version

30.0.0

Implementation Notes

The SSL switch operates asynchronously, thus when a script executes the SSL::enable or SSL::disable commands and then SSL::mode is used, the SSL mode is not yet updated and the previous mode is returned.

SSL::sessionid

Description

Retrieves the connections SSL session ID from the session cache.

Full Syntax

```
SSL::sessionid
```

Valid Events

CLIENTSSL_CLIENTCERT
CLIENT_DATA
CLIENT_CLOSED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
LB_SELECTED
LB_FAILED
PERSIST_DOWN

First Implemented Version

30.0.0

SSL::verify_result

Description

Gets/sets the result code for peer certificate verification.

Full Syntax

```
SSL::verify_result [<result code>]
```

Valid Events

CLIENTSSL_CLIENTCERT
CLIENT_DATA
CLIENT_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
LB_SELECTED
LB_FAILED

PERSIST_DOWN

First Implemented Version
30.0.0

SSL::reject

Description
Rejects an SSL handshake.

Full Syntax
```
SSL::reject
```

Valid Events
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

SSL::sni

Description
Retrieves or changes SNI value
- Retrieves SNI from frontend Client SSL Hello
- Sets SNI value for backend Client SSL Hello

Input
hostname (o)

Output
hostname

Full Syntax
```
SSL::sni [set <hostname>]
```

Valid Events
Get command: HTTP_REQUEST, HTTP_REQUEST_DATA, HTTP_CRULE_MATCH, HTTP_CRULE_NOMATCH
Set command: SERVERSSL_CLIENTHELLO_SEND

First Implemented Version
30.0.3

SSL::version

Description

Retrieves or changes SSL version.

Retrieves the SSL version agreed during frontend SSL handshake

Sets the SSLversions to be included in backend Client Hello.

Input

List of versions (o).

The list can be any continuous combination of the versions required, separated by colon (example: TLSV1_1:TLSV1_2). The input is case insensitive. Versions that are not mentioned are disabled. The following version strings are supported:

SSLV3

TLSV1_0

TLSV1_1

TLSV1_2

TLSV1_3

Output

String

Full Syntax

SSL::version [<version1:version2:...>]

Valid Events

Get command:

CLIENTSSL_CLIENTCERT

CLIENT_DATA

CLIENT_CLOSED

SERVER_DATA

SERVER_CLOSED

HTTP_REQUEST

HTTP_REQUEST_DATA

CRULE_MATCH

CRULE_NOMATCH

HTTP_RESPONSE

HTTP_RESPONSE_DATA

HTTP_RESPONSE_CONTINUE

LB_SELECTED

LB_FAILED

PERSIST_DOWN

Set command:

SERVERSSL_CLIENTHELLO_SEND

First Implemented Version

Get command: 30.2.0.0

Set command: 34.5.4.0

Implementation Notes

Set command not applicable to the special images that support Chinese SM ciphers

TCP Commands

TCP commands provide access to TCP parameters and payloads.

This section describes the following TCP commands:

TCP::client_port

Description

Retrieves the value of the TCP port on which the client opens the connection.

Output

An integer.

Full Syntax

```
TCP::client_port
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE
HTTP_CACHE_RESPONSE

TCP::close

Description

Terminates or resets the TCP connection.

Full Syntax

```
TCP::close
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
SERVER_CONNECTED
SERVER_DATA
CLIENT_CLOSED (reset only)
SERVER_CLOSED (reset only)
HTTP_REQUEST

HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_FILTER_MATCH
LB_SELECTED
LB_FAILED

Sub-commands

- close
- reset
- silent
- twoway

close

Description	Terminates the TCP connection to the client and server and sends a FIN message.
Full Syntax	`TCP::close [reset | silent] [-twoway]`
Notes	If only one parameter is specified, it is interpreted as length.

reset

Description	Resets the TCP connection. When called in a CLIENT_CLOSED or SERVER_CLOSED event, this command resets only the specified context connection (either the client or server connection). When called in any other supported event, this command sends an RST message to both the client and server.
Full Syntax	`TCP::close reset`

silent

Description	Closes the TCP connection silently per context (either the client or the server connection) without sending any notification. Useful when closing a large number of connections, which would otherwise cause many FIN, FINACK, and ACK messages to flood the network.
Full Syntax	`TCP::close silent`

twoway

Description	Terminates the TCP connection at both ends. Closes a front-end connection if called from the back end with the same parameter (reset, silent, or FIN). Closes a back-end connection if called from the front end with the same parameter (reset, silent, or FIN). Useful in splitting mode where the front-end connection is separated from back-end connections.
Full Syntax	`TCP::close twoway`

TCP::close_type

Description

Retrieves the cause of the connection close.

Output

1—graceful termination (FIN)

2—reset (RST)

3—AGING

4—ERROR

5—service configuration changed or real server down (SHUTDOWN)

Full syntax

```
TCP::close_type
```

Valid Events

CLIENT_CLOSED

SERVER_CLOSED

TCP::collect

Description

Collects the specified amount of TCP payload data and triggers an appropriate data event (CLIENT_DATA for a client-side connection, SERVER_DATA for a server-side connection).

If the amount of data to collect is not specified, the appropriate event is triggered after each packet, and data is collected until the TCP::release command is sent.

Output

bytes (optional)

Full Syntax

```
TCP::collect [<bytes>][connect-server]
```

Valid Events

CLIENT_ACCEPTED

CLIENT_DATA

SERVER_CONNECTED

SERVER_DATA

Implementation Notes

- The collect command must be called from the CLIENT_ACCEPTED or SERVER_CONNECTED events to trigger the first CLIENT_DATA or SERVER_DATA events respectively.
- The collect command must be called from the DATA event to trigger the next DATA event.
- The DATA event is triggered only when the requested amount of data (in bytes) is available.

- The collected DATA is clear text (as result of a clear text connection, or offload of the SSL connection).

TCP::detach

Description

Releases a back-end connection back to the idle connection pool when a TCP transaction is completed.

Valid Events

ALL_SIP_EVENTS

ALL_DNS_EVENTS

ON_CLIENT_DATA

ON_SERVER_DATA

Implementation Notes

- The command is relevant for non-HTTP TCP services (Basic-SLB, generic SSL, SIP, DNS) and only when TCP multiplexing is enabled.
- If the command is called while there is still data in the buffer (from an earlier collect) the command will fail and the session will be closed.

TCP::local_port

Description

Retrieves the value of the TCP port on which the switch opens or receives the TCP connection, depending on the context—destination port on the client side, or the port created by Alteon at the back-end connection.

Output

An integer.

Full Syntax

```
TCP::local_port
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE

TCP::option

Description

Retrieves or edits specified TCP option values.

Valid Events

CLIENT_ACCEPTED
SERVER_CONNECTED

First Implemented Version

29.0.2

Sub-Commands

- get
- set

get

Description	Retrieves the raw value of the specified TCP option
Input	option number
Output	If the option does not appear in current connection returns null, otherwise returns raw kind value for the option
Full Syntax	`TCP::option get <option>`

set

Description	Edit (set or add) specified TCP option.
Input	option number value
Output	integer
Full Syntax	`TCP::option set <option> <value>`
Notes	• If the TCP option specified already exists in the TCP headers, change it to the specified value, otherwise such an option should be added. • The specified TCP option can be changed or added on the current packet (next) or on all connection packets (all).

TCP::payload

Description

Retrieves the collected TCP payload string, or replaces it with a specified string.

Valid Events

CLIENT_DATA

SERVER_DATA

Implementation Notes

If the packet type is text (such as HTTP or SIP), this command retrieves the data as text. If the packet type is binary (such as DNS), apply the Tcl "binary scan" and "binary format" string commands to handle the payload string.

Sub-commands

- payload
- replace
- length
- find
- find_all
- clear

payload

Description	Retrieves all or part of the collected TCP payload.
Input	offset (optional) length (optional, mandatory if offset is specified)
Output	string
Full Syntax	`TCP::payload [<offset>] [<length>]`
Notes	If only one parameter is specified, it is interpreted as length.

replace

Description	Replaces all or part of the collected TCP payload with specified data.
Input	offset length data
Full Syntax	`TCP::payload <offset> <length> <data>`
Notes	• All replace operations are performed in-line. • The data size may change according to the defined payload length. Subsequent commands, such as TCP::release, refer to the new data size. • If all data is erased (for example, by the TCP::payload 0 [TCP::payload length] "") command, subsequent processing steps are not activated.

length

Description	Retrieves the length of the collected TCP payload, in bytes.
Output	integer
Full Syntax	`TCP::payload length`

find

Description	Searches for the first occurrence of a string in the collected payload and if found, returns its byte length offset within the payload. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns -1 if no occurrence is found.
Options	-nocase
Input	string start_from_offset (optional)
Output	Offset or -1 (integer)

Full Syntax	`TCP::payload find [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [TCP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the TCP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

find_all

Description	Searches for all non-overlapping occurrences of a string in the collected payload. Returns a list of byte-length offsets. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns an empty list if no occurrence is found.
Options	-nocase
Input	string start_from_offset (optional)
Output	A list of occurrences (Tcl list of integers).
Full Syntax	`TCP::payload find_all [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [TCP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the TCP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

clear

Description	Clears the current payload. This is a shortcut for TCP::payload 0 [TCP::payload length] ""
Full Syntax	`TCP::payload clear`

TCP::release

Description

Releases all collected data, or a specified amount of collected data, for further processing.

Input

length (optional)

Output

An integer.

Full Syntax

```
TCP::release [<length>]
```

Valid Events

CLIENT_DATA

SERVER_DATA

Implementation Notes

- The released value can be lower than the specified length when less than specified length is collected).
- The released value can be higher than the specified length. If this is the last release on relevant connection flow (no further collect is requested), all data will be released, regardless of the actual length parameter.
- When no length is specified, all the collected data is released.
- The released amount of bytes can be up to the available payload (see the TCP::payload command).
- Only released data is forwarded to the next processing steps (such as HTTP processing or forwarding to destination).

TCP::remote_port

Description

Retrieves the value of the TCP port on the remote side of the TCP connection, depending on the context—source port on the client side, destination port on the server side.

Output

An integer.

Full Syntax

```
TCP::remote_port
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_REQUEST
HTTP_REQUEST_DATA
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE

TCP::respond

Description

Sends the specified data to the remote side of the connection.

Input

Data

Full Syntax

```
TCP::respond <data>
```

Valid Events

CLIENT_DATA
SERVER_CONNECTED
SERVER_DATA

TCP::server_port

Description

Retrieves the TCP port on the remote side of a server-side connection.

Output

An integer.

Full Syntax

```
TCP::server_port
```

Valid Events

SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED
HTTP_CRULE_MATCH
HTTP_CRULE_NOMATCH
HTTP_RESPONSE
HTTP_RESPONSE_DATA
HTTP_RESPONSE_CONTINUE

Note: Returns 0 when used before the server port is known.

UDP Commands

UDP commands provide access to UDP parameters and payloads.

This section describes the following UDP commands:

UDP::age

Description
Closes session after a specified period.

Input
Timeout in milliseconds.

Output
None

Full Syntax
```
UDP::age <timeout>
```

Valid Events
UDP_SERVERSIDE_EVENTS

UDP::client_port

Description
Retrieves the value of the UDP port on which the client opens the connection (source port on requests, destination port on replies).

Output
An integer.

Full Syntax
```
UDP::client_port
```

Valid Events
CLIENT_ACCEPTED
CLIENT_CLOSED
CLIENT_DATA
SERVER_CLOSED
SERVER_CONNECTED
SERVER_DATA

UDP::drop

Description
Drops the current UDP packet, but does not remove the relevant entry from the session table.

Full Syntax

```
UDP::drop
```

Valid Events

CLIENT_ACCEPTED

CLIENT_DATA

SERVER_DATA

UDP::local_port

Description

Retrieves the value of the UDP port on which the switch opens or receives the UDP connection, depending on the context—destination port on the client side, or the port created by Alteon at the back-end connection.

Output

An integer.

Full Syntax

```
UDP::local_port
```

Valid Events

CLIENT_ACCEPTED

CLIENT_CLOSED

CLIENT_DATA

SERVER_CLOSED

SERVER_CONNECTED

SERVER_DATA

UDP::payload

Description

Retrieves the collected UDP payload string, or replaces it with a specified string.

Valid Events

CLIENT_ACCEPTED

CLIENT_DATA

SERVER_DATA

Implementation Notes

If the packet type is text (such as HTTP or SIP), this command retrieves the data as text. If the packet type is binary (such as DNS), apply the Tcl “binary scan” and “binary format” string commands to handle the payload string.

Sub-commands

payload

Description	Retrieves all or part of the collected UDP payload.
Input	offset (optional) length (optional, mandatory if offset is specified)
Output	string
Full Syntax	`UDP::payload [<offset>][<length>]`
Notes	If only one parameter is specified, it is interpreted as length.

replace

Description	Replaces all or part of the collected UDP payload with specified data. When length is 0, inserts the specified data in the UDP payload.
Input	offset length data
Full Syntax	`UDP::payload replace <offset> <length> <data>`

length

Description	Retrieves the length of the collected UDP payload, in bytes.
Output	integer
Full Syntax	`UDP::payload length`

find

Description	Searches for the first occurrence of a string in the collected payload and if found, returns its byte length offset within the payload. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns -1 if no occurrence is found.
Options	-nocase
Input	string start_from_offset (optional)

Output	Offset or -1 (integer)
Full Syntax	`UDP::payload find [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [UDP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the UDP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

find_all

Description	Searches for all non-overlapping occurrences of a string in the collected payload. Returns a list of byte-length offsets. You can specify where the search starts (offset), and if the match is case-sensitive or case-insensitive. Returns an empty list if no occurrence is found.
Options	-nocase
Input	string start_from_offset (optional)
Output	A list of occurrences (Tcl list of integers).
Full Syntax	`UDP::payload find_all [-nocase] <string> [<start_from_offset>]`
Notes	• This command is much faster than a native Tcl string command working on [UDP::payload]. The returned offset is byte-length, not string-length, so it is convenient to use with the UDP:: payload replace command. • A case-insensitive search is performed on ASCII characters only.

clear

Description	Clears the current payload. This is a shortcut for UDP::payload 0 [UDP::payload length] ""
Full Syntax	`UDP::payload clear`

UDP::remote_port

Description

Retrieves the value of the UDP port on the remote side of the UDP connection, depending on the context—source port on the client side, destination port on the server side.

Output

An integer.

Full Syntax

```
UDP::remote_port
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
CLIENT_CLOSED
SERVER_CONNECTED
SERVER_DATA
SERVER_CLOSED

UDP::respond

Description

Sends the specified data to the remote side of the connection.

Input

Data

Full Syntax

```
UDP::respond <data>
```

Valid Events

CLIENT_ACCEPTED
CLIENT_DATA
SERVER_DATA

UDP::server_port

Description

Retrieves the UDP port on the remote side of a server-side connection.

Output

An integer.

Full Syntax

```
UDP::server_port
```

Valid Events

SERVER_CLOSED
SERVER_CONNECTED
SERVER_DATA

X509 Commands

This section describes the following X509 commands:

X509::extensions

Description

Retrieves the specified certificate extensions, if they exist.

Input

Certificate

Output

String

Full Syntax

```
X509::extensions [<certificate>]
```

Valid Events

SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version

30.0.0

Implementation Notes

If the certificate is not valid, the script fails with an error.

X509::has_expired

Description

Verifies whether a certificate has expired, or is not yet valid.

Input
Certificate in der encoding.

Output
1 if the certificate has expired or is not yet valid, 0 if valid.

Full Syntax
```
x509::has_expired<certificate>
```

Valid Events
All except
CLIENT_ACCEPTED
SERVER_CONNECTED
CACHE_RESPONSE

First Implemented Version
31.0.1

X509::hash

Description
Performs MD5 hashing on the specified certificate.

Input
Certificate

Output
String

Full Syntax
```
X509::hash [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA

CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::issuer

Description
Retrieves the issuer field from the specified certificate.

Input
Certificate

Output
String

Full Syntax
```
X509::issuer [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

Implementation Notes
Returns the issuer field in the format "C=ZA, ST=Western Cape, L=Cape Town ...".

X509::not_valid_after

Description
Retrieves the end_of_validity_date from the not_valid_after field from the specified certificate.

Input
Certificate

Output
String

Full Syntax
```
X509::not_valid_after [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::not_valid_before

Description
Retrieves the validity_start_date from the not_valid_before field from the specified certificate.

Input
Certificate

Output
String

Full Syntax
```
X509::not_valid_before [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::serial_number

Description
Retrieves the serial_number field from the specified certificate.

Input
Certificate

Output
String

Full Syntax

```
X509::serial_number [<certificate>]
```

Valid Events

SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version

30.0.0

X509::signature_algorithm

Description

Retrieves the signature_algorithm for the specified certificate.

Input

Certificate

Output

String

Full Syntax

```
X509::signature_algorithm [<certificate>]
```

Valid Events

SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA

SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::subject

Description
Retrieves the subject field from the specified certificate.

Input
Certificate

Output
String

Full Syntax
```
X509::subject [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA

CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

Implementation Notes
Returns the subject field in the format "C=ZA, ST=Western Cape, L=Cape Town ...".

X509::subject_public_key

Description
Retrieves the subject_public_key field from the specified certificate.

Input
Certificate

Output
Multi-line value, hex value separated by ":".

Full Syntax
```
X509::subject_public_key [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA

CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::subject_public_key_type

Description
Retrieves the type of the subject_public_key field from the specified certificate.

Input
Certificate

Output
rsaEncryption

Full Syntax
```
X509::subject_public_key_type [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::subject_public_key_RSA_bits

Description

Returns the size, in bits, of the subject public RSA key of the specified X509 certificate.

This command is only applicable when the public key type is RSA. Otherwise, the command returns "unknown".

Input

Certificate

Output

string

Full Syntax

```
X509::subject_public_key_RAS_bits [<certificate>]
```

Valid Events

SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version

30.0.0

X509::verify_cert_error_string

Description

Translates the certificate verification error code into the error string, as per OpenSSL.

Input

Certificate

Output
String

Full Syntax
```
X509::verify_cert_error_string <error code>
```

Valid Events
All

First Implemented Version
30.0.0

X509::verify_signature

Description
Verifies that the given certificate has signed on the given data.

Input
Certificate
The encoding for the certificate—pem or der
The hash algorithm
Signature—The digital signature of the hash function.
Data—The data on which the certificate has signed.

Output
1 if the certificate was verified, 0 if not.

Full Syntax
```
X509::verify_signature <certificate> <der|pem> <hash algo> <signature>
<data>
```

Valid Events
All except
CLIENT_ACCEPTED
SERVER_CONNECTED
CACHE_RESPONSE

First Implemented Version
31.0.1

X509::version

Description

Retrieves the version of the specified certificate.

Input
Certificate

Output
Integer

Full Syntax
```
X509::version [<certificate>]
```

Valid Events
SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version
30.0.0

X509::whole

Description
Retrieves the entire specified certificate, in PEM format.

Input
Certificate

Output
Integer

Full Syntax

```
X509::whole [<certificate>]
```

Valid Events

SIP_REQUEST
PERSIST_DOWN
CLIENT_DATA
SERVER_DATA
CLIENT_CLOSED
SERVER_CLOSED
NAME_RESOLVED
LB_SELECTED
LB_FAILED
HTTP_REQUEST
HTTP_REQUEST_DATA
CONTENT_RULE_MATCH
CONTENT_RULE_NOMATCH
HTTP_RESPONSE
HTTP_100_CONTINUE
HTTP_RESPONSE_DATA
CLIENTSSL_CLIENTCERT

First Implemented Version

30.0.0

Operators

This section describes the following command operators:

equals

Description

An operator for string comparison that checks if two strings are equal.

Output

1 (for true) or 0 (for false).

Full Syntax

```
<string1> equals <string2>
```

First Implemented Version

29.3.0 for the global class command, 30.0.0 for generic string commands.

contains

Description

An operator for string comparison that checks if a string contains the sub-string. Available only with the global class command.

Output

1 (for true) or 0 (for false).

Full Syntax

```
<string> contains <sub-string>
```

First Implemented Version

29.3.0

starts_with

Description

An operator for string comparison that checks if a string starts with the specified sub-string. Available only with the global class command.

Output

1 (for true) or 0 (for false).

Full Syntax

```
<string> starts_with <sub-string>
```

First Implemented Version

29.3.0

ends_with

Description

An operator for string comparison that checks if a string ends with the specified sub-string. Available only with the global class command.

Output

1 (for true) or 0 (for false).

Full Syntax

```
<string> ends_with <sub-string>
```

First Implemented Version

29.3.0

CHAPTER 3 – APPSHAPE++ EVENTS

This section describes the events that AppShape++ supports.

The following events are supported:

- INIT
- CLIENT_ACCEPTED
- CLIENT_DATA
- CLIENT_CLOSED
- CLIENTSSL_CLIENTCERT
- SERVER_CONNECTED
- SERVER_DATA
- SERVER_CLOSED
- SERVERSSL_CLIENTHELLO_SEND
- DNS_REQUEST
- DNS_RESPONSE
- DNS_VALIDATION_FAILED
- HTTP_CACHE_RESPONSE
- HTTP_CRULE_MATCH
- HTTP_CRULE_NOMATCH
- HTTP_REQUEST
- HTTP_REQUEST_DATA
- HTTP_RESPONSE
- HTTP_RESPONSE_DATA
- HTTP_RESPONSE_CONTINUE
- HTTP_FILTER_MATCH
- LB_SELECTED
- LB_FAILED
- PERSIST_DOWN
- SIDEBAND_FAILURE
- SIDEBAND_RESPONSE
- SIP_REQUEST
- SIP_RESPONSE

INIT

Description

This event is triggered when an AppShape++ script is added to the configuration, or is modified. During this event, static variables should be declared.

First Implemented Version

29.0.0

Alias

RULE_INIT

CLIENT_ACCEPTED

Description

This event is triggered when a client establishes a new connection.

- For TCP, the event is triggered when the TCP handshake is completed.
- For UDP, the event is triggered when a new UDP session arrives (source IP/port and destination IP/port that does not yet exist in the session table).

First Implemented Version

29.0.0

CLIENT_DATA

Description

This event is triggered when client data is received.

- For TCP, the event is triggered when new data arrives from the client, when data collection is requested using the TCP::collect command.
- For UDP, the event is triggered for each UDP datagram.

First Implemented Version

29.0.0

CLIENT_CLOSED

Description

This event is triggered when the client-side connection closes. A connection closes when a connection close exchange occurs (TCP only), or there is no activity for the defined timeout.

For UDP, the event is triggered after session aging.

First Implemented Version

29.0.0

CLIENTSSL_CLIENTCERT

Description

This event is triggered when a certificate is received from the client during the SSL handshake.

First Implemented Version

30.0.0

SERVER_CONNECTED

Description

- For TCP, this event is triggered when a TCP connection is established with the back-end server.
- For UDP, this event is triggered when a first UDP datagram is about to be sent to the selected server.

First Implemented Version

29.0.0

SERVER_DATA

Description

This event is triggered when data is received from the back-end server.

- For TCP, the event is triggered when new data arrives from the server, when data collection is requested using the TCP::collect command on the back-end connection.
- For UDP, the event is triggered for each UDP datagram received from the server.

First Implemented Version
29.0.0

SERVER_CLOSED

Description
This event is triggered when the server-side connection closes. A connection closes when a connection close exchange occurs (TCP only), or there is no activity for the defined timeout.

For UDP, the event is triggered after session aging.

First Implemented Version
29.0.0

SERVERSSL_CLIENTHELLO_SEND

Description
This event is triggered just before the Client SSL Hello message is sent to server, allowing to set the SNI value for the backend connectio..

First Implemented Version
31.0.3

DNS_REQUEST

Description
This event is triggered when a DNS request is received.

First Implemented Version
30.2.0

DNS_RESPONSE

Description
This event is triggered when a DNS response is received.

First Implemented Version
30.2.0

DNS_VALIDATION_FAILED

Description
This event is triggered when the validation of a DNS request or response has failed.

First Implemented Version

30.2.0

HTTP_CACHE_RESPONSE

Description

This event is triggered when the requested HTTP page is served from the cache, and is allowed to overwrite the response before sending the HTTP page to the client.

First Implemented Version

29.0.0

HTTP_CRULE_MATCH

Description

This event is triggered when an HTTP request matches one of the virtual service content-based rules.

First Implemented Version

29.0.0

Alias

HTTP_CLASS_SELECTED

HTTP_CRULE_NOMATCH

Description

This event is triggered when an HTTP request does not match any of the virtual service content-based rules.

First Implemented Version

29.0.0

Alias

HTTP_CLASS_FAILED

HTTP_REQUEST

Description

This event is triggered when an HTTP request is received after all message headers are parsed, and before headers are streamed.

First Implemented Version

29.0.0

HTTP_REQUEST_DATA

Description

This event is triggered during HTTP request payload collection (after the HTTP::collect command is invoked on the front-end connection), when the requested amount of data is gathered, or the end of the message is detected.

First Implemented Version

29.0.0

HTTP_RESPONSE

Description

This event is triggered when an HTTP response is received, after all message headers are parsed and before headers are streamed.

This event is not triggered when a 100 Continue message is received from the server., or when the requested HTTP page is served from the cache.

First Implemented Version

29.0.0

HTTP_RESPONSE_DATA

Description

This event is triggered during HTTP response payload collection (after the HTTP::collect command is invoked on the back-end connection), when the requested amount of data is gathered, or the end of the message is detected.

First Implemented Version

29.0.0

HTTP_RESPONSE_CONTINUE

Description

This event is triggered when a 100 Continue message is received from the server.

First Implemented Version

29.0.0

HTTP_FILTER_MATCH

Description

This event is triggered on an HTTP filter under the following conditions:

- On the first HTTP request that matches the filter if *Parse All Packets* is disabled.
- On each HTTP request that matches the filter if *Parse All Packets* is enabled.

First Implemented Version

29.3.0

LB_SELECTED

Description

This event is triggered when a real server is selected.

First Implemented Version

29.3.0

LB_FAILED

Description

This event is triggered when real server selection fails.

Real server selection can fail due to following reasons:

- All real servers are unavailable (administratively disabled, operationally disabled, or failed health checks).
- All real servers are overloaded (the real server connection limit has been reached).

First Implemented Version

29.3.0

PERSIST_DOWN

Description

Triggered when the server allocated to a persistent session is down, even when an alternate server is available.

This event is valid only for persistent sessions managed via the AppShape++ persist command.

First Implemented Version

29.4.0

SIDEBAND_FAILURE

Description

Triggered when a sideband encounters a problem that prevents it from returning a valid response.

This event can occur in instances such as:

- Backend server timeout
- Shortage of resources
- Loss of internal communication between sessions

First Implemented Version
33.0.1.50

SIDEBAND_RESPONSE

Description
Triggered when a response message arrives on the sideband channel.

First Implemented Version
33.0.1.50

SIP_REQUEST

Description
Triggered when all headers of a SIP request are received.

First Implemented Version
29.4.0

SIP_RESPONSE

Description
Triggered when all headers of a SIP response are received.

First Implemented Version
29.4.0

APPENDIX A – SUPPORTED TCL COMMANDS

This appendix lists the Tcl commands supported by AppShape++. For descriptions of these commands, and a complete list of Tcl commands, see www.tcl.tk/man/tcl8.4/TclCmd/contents.htm.

append	eval	join	lset	split
array	expr	lappend	lsort	string
binary	for	lindex	namespace	switch
break	foreach	linsert	regexp	time
catch	format	list	regsub	unknown
clock	global	llength	regsource	unset
concat	if	lrange	return	variable
continue	incr	lreplace	scan	while
error	info exists	lsearch	set	

RADWARE LTD. END USER LICENSE AGREEMENT

By accepting this End User License Agreement (this "License Agreement") you agree to be contacted by Radware Ltd.'s ("Radware") sales personnel.

If you would like to receive license rights different from the rights granted below or if you wish to acquire warranty or support services beyond the scope provided herein (if any), please contact Radware's sales team.

THIS LICENSE AGREEMENT GOVERNS YOUR USE OF ANY SOFTWARE DEVELOPED AND/OR DISTRIBUTED BY RADWARE AND ANY UPGRADES, MODIFIED VERSIONS, UPDATES, ADDITIONS, AND COPIES OF THE SOFTWARE FURNISHED TO YOU DURING THE TERM OF THE LICENSE GRANTED HEREIN (THE "SOFTWARE"). THIS LICENSE AGREEMENT APPLIES REGARDLESS OF WHETHER THE SOFTWARE IS DELIVERED TO YOU AS AN EMBEDDED COMPONENT OF A RADWARE PRODUCT ("PRODUCT"), OR WHETHER IT IS DELIVERED AS A STANDALONE SOFTWARE PRODUCT. FOR THE AVOIDANCE OF DOUBT IT IS HEREBY CLARIFIED THAT THIS LICENSE AGREEMENT APPLIES TO PLUG-INS, CONNECTORS, EXTENSIONS AND SIMILAR SOFTWARE COMPONENTS DEVELOPED BY RADWARE THAT CONNECT OR INTEGRATE A RADWARE PRODUCT WITH THE PRODUCT OF A THIRD PARTY (COLLECTIVELY, "CONNECTORS") FOR PROVISIONING, DECOMMISSIONING, MANAGING, CONFIGURING OR MONITORING RADWARE PRODUCTS. THE APPLICABILITY OF THIS LICENSE AGREEMENT TO CONNECTORS IS REGARDLESS OF WHETHER SUCH CONNECTORS ARE DISTRIBUTED TO YOU BY RADWARE OR BY A THIRD PARTY PRODUCT VENDOR. IN CASE A CONNECTOR IS DISTRIBUTED TO YOU BY A THIRD PARTY PRODUCT VENDOR PURSUANT TO THE TERMS OF AN AGREEMENT BETWEEN YOU AND THE THIRD PARTY PRODUCT VENDOR, THEN, AS BETWEEN RADWARE AND YOURSELF, TO THE EXTENT THERE IS ANY DISCREPANCY OR INCONSISTENCY BETWEEN THE TERMS OF THIS LICENSE AGREEMENT AND THE TERMS OF THE AGREEMENT BETWEEN YOU AND THE THIRD PARTY PRODUCT VENDOR, THE TERMS OF THIS LICENSE AGREEMENT WILL GOVERN AND PREVAIL. PLEASE READ THE TERMS AND CONDITIONS OF THIS LICENSE AGREEMENT CAREFULLY BEFORE OPENING THE PACKAGE CONTAINING RADWARE'S PRODUCT, OR BEFORE DOWNLOADING, INSTALLING, COPYING OR OTHERWISE USING RADWARE'S STANDALONE SOFTWARE (AS APPLICABLE). THE SOFTWARE IS LICENSED (NOT SOLD). BY OPENING THE PACKAGE CONTAINING RADWARE'S PRODUCT, OR BY DOWNLOADING, INSTALLING, COPYING OR USING THE SOFTWARE (AS APPLICABLE), YOU CONFIRM THAT YOU HAVE READ AND UNDERSTAND THIS LICENSE AGREEMENT AND YOU AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE AGREEMENT. FURTHERMORE, YOU HEREBY WAIVE ANY CLAIM OR RIGHT THAT YOU MAY HAVE TO ASSERT THAT YOUR ACCEPTANCE AS STATED HEREINABOVE IS NOT THE EQUIVALENT OF, OR DEEMED AS, A VALID SIGNATURE TO THIS LICENSE AGREEMENT. IF YOU ARE NOT WILLING TO BE BOUND BY THE TERMS OF THIS LICENSE AGREEMENT, YOU SHOULD PROMPTLY RETURN THE UNOPENED PRODUCT PACKAGE OR YOU SHOULD NOT DOWNLOAD, INSTALL, COPY OR OTHERWISE USE THE SOFTWARE (AS APPLICABLE). THIS LICENSE AGREEMENT REPRESENTS THE ENTIRE AGREEMENT CONCERNING THE SOFTWARE BETWEEN YOU AND RADWARE, AND SUPERSEDES ANY AND ALL PRIOR PROPOSALS, REPRESENTATIONS, OR UNDERSTANDINGS BETWEEN THE PARTIES. "YOU" MEANS THE NATURAL PERSON OR THE ENTITY THAT IS AGREEING TO BE BOUND BY THIS LICENSE AGREEMENT, THEIR EMPLOYEES AND THIRD PARTY CONTRACTORS. YOU SHALL BE LIABLE FOR ANY FAILURE BY SUCH EMPLOYEES AND THIRD PARTY CONTRACTORS TO COMPLY WITH THE TERMS OF THIS LICENSE AGREEMENT.

1. **License Grant**. Subject to the terms of this Agreement, Radware hereby grants to you, and you accept, a limited, nonexclusive, nontransferable license to install and use the Software in machine-readable, object code form only and solely for your internal business purposes ("Commercial License"). If the Software is distributed to you with a software development kit (the "SDK"), then, solely with regard to the SDK, the Commercial License above also includes a limited, nonexclusive, nontransferable license to install and use the SDK solely on computers within your organization, and solely for your internal development of an integration or interoperation of the Software and/or other Radware Products with software or hardware products owned, licensed and/or controlled by you (the "SDK Purpose"). To the extent an SDK is

distributed to you together with code samples in source code format (the "Code Samples") that are meant to illustrate and teach you how to configure, monitor and/or control the Software and/or any other Radware Products, the Commercial License above further includes a limited, nonexclusive, nontransferable license to copy and modify the Code Samples and create derivative works based thereon solely for the SDK Purpose and solely on computers within your organization. The SDK shall be considered part of the term "Software" for all purposes of this License Agreement. You agree that you will not sell, assign, license, sublicense, transfer, pledge, lease, rent or share your rights under this License Agreement nor will you distribute copies of the Software or any parts thereof. Rights not specifically granted herein, are specifically prohibited.

2. **Evaluation Use**. Notwithstanding anything to the contrary in this License Agreement, if the Software is provided to you for evaluation purposes, as indicated in your purchase order or sales receipt, on the website from which you download the Software, as inferred from any time-limited evaluation license keys that you are provided with to activate the Software, or otherwise, then You may use the Software only for internal evaluation purposes ("Evaluation Use") for a maximum of 30 days or such other duration as may specified by Radware in writing at its sole discretion (the "Evaluation Period"). The evaluation copy of the Software contains a feature that will automatically disable it after expiration of the Evaluation Period. You agree not to disable, destroy, or remove this feature of the Software, and any attempt to do so will be a material breach of this License Agreement. During or at the end of the evaluation period, you may contact Radware sales team to purchase a Commercial License to continue using the Software pursuant to the terms of this License Agreement. If you elect not to purchase a Commercial License, you agree to stop using the Software and to delete the evaluation copy received hereunder from all computers under your possession or control at the end of the Evaluation Period. In any event, your continued use of the Software beyond the Evaluation Period (if possible) shall be deemed your acceptance of a Commercial License to the Software pursuant to the terms of this License Agreement, and you agree to pay Radware any amounts due for any applicable license fees at Radware's then-current list prices.
3. **Lab/Development License**. Notwithstanding anything to the contrary in this License Agreement, if the Software is provided to you for use in your lab or for development purposes, as indicated in your purchase order, sales receipt, the part number description for the Software, the Web page from which you download the Software, or otherwise, then You may use the Software only in your lab and only in connection with Radware Products that you purchased or will purchase (in case of a lab license) or for internal testing and development purposes (in case of a development license) but not for any production use purposes.
4. **Subscription Software**. If you licensed the Software on a subscription basis, your rights to use the Software are limited to the subscription period. You have the option to extend your subscription. If you extend your subscription, you may continue using the Software until the end of your extended subscription period. If you do not extend your subscription, after the expiration of your subscription, you are legally obligated to discontinue your use of the Software and completely remove the Software from your system.
5. **Feedback**. Any feedback concerning the Software including, without limitation, identifying potential errors and improvements, recommended changes or suggestions ("**Feedback**"), provided by you to Radware will be owned exclusively by Radware and considered Radware's confidential information. By providing Feedback to Radware, you hereby assign to Radware all of your right, title and interest in any such Feedback, including all intellectual property rights therein. With regard to any rights in such Feedback that cannot, under applicable law, be assigned to Radware, you hereby irrevocably waives such rights in favor of Radware and grants Radware under such rights in the Feedback, a worldwide, perpetual royalty-free, irrevocable, sub-licensable and non-exclusive license, to use, reproduce, disclose, sublicense, modify, make, have made, distribute, sell, offer for sale, display, perform, create derivative works of and otherwise exploit the Feedback without restriction. The provisions of this Section 5 will survive the termination or expiration of this Agreement.
6. **Limitations on Use**. You agree that you will not: (a) copy, modify, translate, adapt or create any derivative works based on the Software; or (b) sublicense or transfer the Software, or include the Software or any portion thereof in any product; or (b) reverse assemble, disassemble, decompile, reverse engineer or otherwise attempt to derive source code (or the

underlying ideas, algorithms, structure or organization) from the Software, in whole or in part, except and only to the extent: (i) applicable law expressly permits any such action despite this limitation, in which case you agree to provide Radware at least ninety (90) days advance written notice of your belief that such action is warranted and permitted and to provide Radware with an opportunity to evaluate if the law's requirements necessitate such action, or (ii) required to debug changes to any third party LGPL-libraries linked to by the Software; or (c) create, develop, license, install, use, or deploy any software or services to circumvent, enable, modify or provide access, permissions or rights which violate the technical restrictions of the Software; (d) in the event the Software is provided as an embedded or bundled component of another Radware Product, you shall not use the Software other than as part of the combined Product and for the purposes for which the combined Product is intended; (e) remove any copyright notices, identification or any other proprietary notices from the Software (including any notices of Third Party Software (as defined below); or (f) copy the Software onto any public or distributed network or use the Software to operate in or as a time-sharing, outsourcing, service bureau, application service provider, or managed service provider environment. Notwithstanding the foregoing, if you provide hosting or cloud computing services to your customers, you are entitled to use and include the Software in your IT infrastructure on which you provide your services. Lastly, if you acquire Software under Radware's Global Elastic License (GEL) model, you commit to use any such Software only as an Alteon VA on COTS server or on GEL-dedicated hardware platforms as indicated in the part description of such hardware (be it hardware originally purchased as GEL-dedicated or later upgraded to be GEL-dedicated). Use of Software under a GEL model on a non-GEL-dedicated hardware platform is prohibited. If you deploy GEL model Software on a virtual platform, you can do so without the virtual platform being GEL-dedicated. It is hereby clarified that the prohibitions on modifying, or creating derivative works based on, any Software provided by Radware, apply whether the Software is provided in a machine or in a human readable form. Human readable Software to which this prohibition applies includes (without limitation) "Radware AppShape++ Script Files" that contain "Special License Terms". It is acknowledged that examples provided in a human readable form may be modified by a user.

7. **Intellectual Property Rights**. You acknowledge and agree that this License Agreement does not convey to you any interest in the Software except for the limited right to use the Software, and that all right, title, and interest in and to the Software, including any and all associated intellectual property rights, are and shall remain with Radware or its third party licensors. You further acknowledge and agree that the Software is a proprietary product of Radware and/or its licensors and is protected under applicable copyright law.

8. **No Warranty**. The Software, and any and all accompanying software, files, libraries, data and materials, are distributed and provided "AS IS" by Radware or by its third party licensors (as applicable) and with no warranty of any kind, whether express or implied, including, without limitation, any non-infringement warranty or warranty of merchantability or fitness for a particular purpose. Neither Radware nor any of its affiliates or licensors warrants, guarantees, or makes any representation regarding the title in the Software, the use of, or the results of the use of the Software. Neither Radware nor any of its affiliates or licensors warrants that the operation of the Software will be uninterrupted or error-free, or that the use of any passwords, license keys and/or encryption features will be effective in preventing the unintentional disclosure of information contained in any file. You acknowledge that good data processing procedure dictates that any program, including the Software, must be thoroughly tested with non-critical data before there is any reliance on it, and you hereby assume the entire risk of all use of the copies of the Software covered by this License. Radware does not make any representation or warranty, nor does Radware assume any responsibility or liability or provide any license or technical maintenance and support for any operating systems, databases, migration tools or any other software component provided by a third party supplier and with which the Software is meant to interoperate.

This disclaimer of warranty constitutes an essential and material part of this License.

In the event that, notwithstanding the disclaimer of warranty above, Radware is held liable under any warranty provision, Radware shall be released from all such obligations in the event that the Software shall have been subject to misuse, neglect, accident or improper installation, or if repairs or modifications were made by persons other than by Radware's authorized service personnel.

9. **Limitation of Liability**. Except to the extent expressly prohibited by applicable statutes, in no event shall Radware, or its principals, shareholders, officers, employees, affiliates, licensors, contractors, subsidiaries, or parent organizations (together, the "Radware Parties"), be liable for any direct, indirect, incidental, consequential, special, or punitive damages whatsoever relating to the use of, or the inability to use, the Software, or to your relationship with, Radware or any of the Radware Parties (including, without limitation, loss or disclosure of data or information, and/or loss of profit, revenue, business opportunity or business advantage, and/or business interruption), whether based upon a claim or action of contract, warranty, negligence, strict liability, contribution, indemnity, or any other legal theory or cause of action, even if advised of the possibility of such damages. If any Radware Party is found to be liable to You or to any third-party under any applicable law despite the explicit disclaimers and limitations under these terms, then any liability of such Radware Party, will be limited exclusively to refund of any license or registration or subscription fees paid by you to Radware.
10. **Third Party Software**. The Software includes software portions developed and owned by third parties (the "Third Party Software"). Third Party Software shall be deemed part of the Software for all intents and purposes of this License Agreement; provided, however, that in the event that a Third Party Software is a software for which the source code is made available under an open source software license agreement, then, to the extent there is any discrepancy or inconsistency between the terms of this License Agreement and the terms of any such open source license agreement (including, for example, license rights in the open source license agreement that are broader than the license rights set forth in Section 1 above and/or no limitation in the open source license agreement on the actions set forth in Section 6 above), the terms of any such open source license agreement will govern and prevail. The terms of open source license agreements and copyright notices under which Third Party Software is being licensed to Radware or a link thereto, are included with the Software documentation or in the header or readme files of the Software. Third Party licensors and suppliers retain all right, title and interest in and to the Third Party Software and all copies thereof, including all copyright and other intellectual property associated therewith. In addition to the use limitations applicable to Third Party Software pursuant to Section 6 above, you agree and undertake not to use the Third Party Software as a general SQL server, as a stand-alone application or with applications other than the Software under this License Agreement.
11. **Term and Termination**. This License Agreement is effective upon the first to occur of your opening the package of the Product, purchasing, downloading, installing, copying or using the Software or any portion thereof, and shall continue until terminated. However, sections 5-15 shall survive any termination of this License Agreement. The Licenses granted under this License Agreement are not transferable and will terminate upon: (i) termination of this License Agreement, or (ii) transfer of the Software, or (iii) in the event the Software is provided as an embedded or bundled component of another Radware Product, when the Software is unbundled from such Product or otherwise used other than as part of such Product. If the Software is licensed on subscription basis, this Agreement will automatically terminate upon the termination of your subscription period if it is not extended.
12. **Export**. The Software or any part thereof may be subject to export or import controls under applicable export/import control laws and regulations including such laws and regulations of the United States and/or Israel. You agree to comply with such laws and regulations, and, agree not to knowingly export, re-export, import or re-import, or transfer products without first obtaining all required Government authorizations or licenses therefor. Furthermore, You hereby covenant and agree to ensure that your use of the Software is in compliance with all other foreign, federal, state, and local laws and regulations, including without limitation all laws and regulations relating to privacy rights, and data protection. You shall have in place a privacy

policy and obtain all of the permissions, authorizations and consents required by applicable law for use of cookies and processing of users' data (including without limitation pursuant to Directives 95/46/EC, 2002/58/EC and 2009/136/EC of the EU if applicable) for the purpose of provision of any services.

13. **US Government**. To the extent you are the U.S. government or any agency or instrumentality thereof, you acknowledge and agree that the Software is a "commercial computer software" and "commercial computer software documentation" pursuant to applicable regulations and your use of the Software is subject to the terms of this License Agreement.
14. **Federal Acquisition Regulation (FAR)/Data Rights Notice**. Radware's commercial computer software is created solely at private expense and is subject to Radware's commercial license rights.
15. **Governing Law**. This License Agreement shall be construed and governed in accordance with the laws of the State of Israel.
16. **Miscellaneous**. If a judicial determination is made that any of the provisions contained in this License Agreement is unreasonable, illegal or otherwise unenforceable, such provision or provisions shall be rendered void or invalid only to the extent that such judicial determination finds such provisions to be unreasonable, illegal or otherwise unenforceable, and the remainder of this License Agreement shall remain operative and in full force and effect. In any event a party breaches or threatens to commit a breach of this License Agreement, the other party will, in addition to any other remedies available to, be entitled to injunction relief. This License Agreement constitutes the entire agreement between the parties hereto and supersedes all prior agreements between the parties hereto with respect to the subject matter hereof. The failure of any party hereto to require the performance of any provisions of this License Agreement shall in no manner affect the right to enforce the same. No waiver by any party hereto of any provisions or of any breach of any provisions of this License Agreement shall be deemed or construed either as a further or continuing waiver of any such provisions or breach waiver or as a waiver of any other provision or breach of any other provision of this License Agreement.

IF YOU DO NOT AGREE WITH THE TERMS OF THIS LICENSE YOU MUST REMOVE THE SOFTWARE FROM ANY DEVICE OWNED BY YOU AND IMMEDIATELY CEASE USING THE SOFTWARE.