ATSC PS/100-6
05 Nov 2002
Revision 2

DTV APPLICATION SOFTWARE ENVIRONMENT LEVEL 1 (DASE-1)
PART 6: SECURITY

ATSC Approved Proposed Standard

ADVANCED TELEVISION SYSTEMS COMMITTEE

Blank Page

Table of Contents

DASE-1 SECURITY 1
1. SCOPE 1
1.1 Status 1
1.2 Purpose 1
1.3 Application 1
1.4 Organization 2
2. REFERENCES 3
2.1 Normative References 3
2.2 Informative References 3
2.3 Reference Acquisition 4
3. DEFINITIONS 5
3.1 Conformance Keywords 5
3.2 Acronyms and Abbreviations 5
3.3 Terms 5
4. BEHAVIOR 6
4.1 Authorization 6
4.2 Runtime Code Extension 7
5. FACILITIES 9
5.1 Permission Authorization Content 9
5.1.1 application/dase-permission 9
ANNEX A. DOCUMENT TYPE DEFINITIONS 12
A.1 Permission Authorization Document Type 12
ANNEX B. PRIVILEGED OPERATIONS 14
B.1 Declarative Application Environment 14
B.2 Procedural Application Environment 14
ANNEX C. PERMISSION AUTHORIZATIONS 16
C.1 Cookie 16
C.2 DisplayConfig 16
C.3 File 16
C.4 MediaSelect 17
C.5 Preference 17
C.6 Property 17
C.7 RuntimeCodeExtension 17
C.8 Select 17
C.9 ServiceContext 18
C.10 ServiceInfoAccess 18
C.11 Socket 18
C.12 StateManagement 18
C.13 User 19
C.14 Xlet 19
ANNEX D. EXAMPLES 20
D.1 Permission Authorization Request Example 20
CHANGES 21
Changes from Candidate Standard to Proposed Standard 21

Table of Tables

Table 1 Declarative Application Environment Privileged Operations 14
Table 2 Procedural Application Environment Privileged Operations 14
Table 3 Changes from Candidate Standard 21

Blank Page

DASE-1 Security

ATSC Approved Proposed Standard

1. SCOPE

1.1 Status

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained by the ATSC.

This specification is an ATSC Approved Proposed Standard, having passed ATSC Member Ballot on September 16, 2002. This document is an editorial revision of the Proposed Standard (PS/100-6) dated August 19, 2002, having incorporated resolutions of comments that occurred during the ATSC Member Ballot.

This specification is expected to be published as ATSC Standard A/100-6 upon the finalization of two specifications normatively referenced by the DASE Standard: (1) the ATSC Application Reference Model (ARM), currently ATSC Approved Proposed Standard PS/94, and (2) the W3C DOM-2 HTML Specification, currently a W3C Candidate Recommendation.

The ATSC believes that this specification is stable, that it has been substantially demonstrated in independent implementations, and that it adequately addresses issues identified during the Candidate Standard phase. A list of cumulative changes made to this specification since the Candidate Standard phase began may be found at the end of this document.

A list of current ATSC Standards and other technical documents can be found at http://www.atsc.org/standards.html.

1.2 Purpose

This specification defines a framework by means of which DASE applications may gain access to privileged operations of declarative and procedural application environments embodied by a compliant receiver.[1]

This specification does not define or rely upon the use of conditional access facilities; furthermore, this specification does not directly support any form of intellectual property management system.

1.3 Application

The behavior and facilities of this specification are intended to apply to terrestrial (over-the-air) broadcast systems and receivers. In addition, the same behavior and facilities may be applied to other transport systems (such as cable or satellite).

[1] The user's attention is called to the possibility that compliance with this standard may require use of an invention covered by patent rights. By publication of this standard, no position is taken with respect to the validity of this claim, or of any patent rights in connection therewith. The patent holder has, however, filed a statement of willingness to grant a license under these rights on reasonable and nondiscriminatory terms and conditions to applicants desiring to obtain such a license. Details may be obtained from the publisher.

1.4 Organization

This specification is organized as follows:

- Section 1 Describes purpose, application and organization of this specification
- Section 2 Enumerates normative and informative references
- Section 3 Defines acronyms, terminology, and conventions
- Section 4 Specifies security framework behavior
- Section 5 Specifies security facilities
- Annex A Specifies document type definitions used by security facilities
- Annex B Specifies privileged operations
- Annex C Specifies permission authorizations
- Annex D Depicts examples of security facilities
- Changes Cumulative changes to specification

Unless explicitly indicated otherwise, all annexes shall be interpreted as normative parts of this specification.

This specification makes use of certain notational devices to provide valuable informative and explanatory information in the context of normative and, occasionally, informative sections. These devices take the form of paragraphs labeled as *Example* or *Note*. In each of these cases, the material is to be considered informative in nature.

2. REFERENCES

This section defines the normative and informative references employed by this specification. With the exception of Section 2.1, this section and its subsections are informative; in contrast, Section 2.1 is normative.

2.1 Normative References

The following documents contain provisions which, through reference in this specification, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All referenced documents are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the referenced document.

Note: This specification uses a reference notation based on acronyms or convenient labels for identifying a reference (as opposed to using numbers).

[DASE]

DASE-1 Part 1: Introduction, Architecture, and Common Facilities, PS/100-1, ATSC

[XML]

Extensible Markup Language (XML) 1.0, Recommendation, W3C

[XMLNAMES]

Namespaces in XML, Recommendation, W3C

2.2 Informative References

[DASE-API]

DASE-1 Part 4: Application Programming Interface, PS/100-4, ATSC

[DASE-ARM]

DASE-1 Part 7: Application Delivery System – ARM Binding, PS/100-7, ATSC

[DASE-DA]

DASE-1 Part 2: Declarative Applications and Environment, PS/100-2, ATSC

[DASE-PA]

DASE-1 Part 3: Procedural Applications and Environment, PS/100-3, ATSC

[JAVATV]

Java TV API Specification, Version 1.0, http://java.sun.com/products/javatv/, Sun Microsystems

[PJAE]

PersonalJava™ Application Environment Specification, Version 1.2A, http://java.sun.com/products/personaljava/, Sun Microsystems

2.3 Reference Acquisition

ATSC Standards

Advanced Television Systems Committee (ATSC), 1750 K Street N.W., Suite 1200 Washington, DC 20006 USA; Phone: +1 202 828 3130; Fax: +1 202 828 3131; http://www.atsc.org/.

Java Standards

Sun Microsystems, Inc., 901 San Antonio Road, Palo Alto, CA 94303 USA; http://java.sun.com/.

W3C Standards

World Wide Web Consortium (W3C), Massachusetts Institute of Technology, Laboratory for Computer Science, 200 Technology Square, Cambridge, MA 02139, USA; Phone: +1 617 253 2613; Fax: +1 617 258 5999; http://www.w3.org/.

3. DEFINITIONS

This section defines conformance keywords, acronyms and abbreviations, and terms as employed by this specification.

All acronyms, abbreviations, and terms defined by [DASE] apply to this specification. Only those acronyms, abbreviations, and terms specific to this document and not common to DASE in its entirety are defined herein.

3.1 Conformance Keywords

As used in this document, the conformance keyword *shall* denotes a mandatory provision of the standard. The keyword *should* denotes a provision that is recommended but not mandatory. The keyword *may* denotes a feature whose presence does not preclude compliance, that may or may not be present at the option of the DASE Application or the DASE System implementer.

3.2 Acronyms and Abbreviations

RCE Runtime Code Extension

3.3 Terms

application emission policy: a set of rules for determining grantable permissions based on conditions defined by the application emitter.

authorization: the process of determining if a privileged operation is permitted.

authorized operation: an unprivileged operation or a privileged operation for which permission has been granted.

effective permissions: a set of granted permissions which apply to an application instance.

grantable permission: a permission which may be granted if requested; determined by permission policy.

granted permission: a requested permission which is granted to a subject.

local policy: a set of rules for determining grantable permissions based on conditions defined by both the receiver manufacturer and the end-user.

permission: authorization to perform a privileged operation.

permission policy: a set of rules for determining grantable permissions.

privileged operation: an operation which is controlled by the security framework.

requested permission: a request for authorization to perform a privileged operation.

security framework: a set of services which provide protection from certain threats.

unprivileged operation: an operation which is not controlled by the security framework.

4. BEHAVIOR

This section describes certain normative behavior for applications and application environments which employ the facilities defined by this specification.

4.1 Authorization

A DASE Application shall be authorized to perform certain privileged operations if and only if the application requests and the DASE System grants permission to perform these operations.

A DASE Application may request permission to perform certain *privileged operation*s designated by ANNEX B. Any operation permitted by a DASE System which is not specified in ANNEX B shall be considered an *unprivileged operation*.

A DASE System shall determine the application's *effective permissions*, also known as the *granted permissions*, by following the procedures specified in Section 4.1.1, Effective Permissions Determination.

The determination of a DASE Application's effective permissions shall occur only once and shall occur prior to processing the application's initial entity. This determination shall remain in effect until the application is terminated.

Each time a DASE System attempts to perform a privileged operation on behalf of a DASE Application, the DASE System shall determine if the operation is authorized (i.e., permitted) by examining the application's *effective permissions*. If the effective permissions grant the privileged operation, the operation shall proceed as normal; otherwise, a runtime security exception shall be raised.

The granting or denying of permissions to privileged operations shall occur no earlier than the moment when the DASE System attempts to perform a privileged operation on behalf of an application. However, a DASE System may determine prior to this point whether a privileged operation will be permitted or denied in the future.

> *Note:* For example, an implementation may determine at the time that an instance of `java.io.File` is constructed that no read or write access will be granted based on the *effective permissions*; however, it does not raise a runtime security exception at this time, but waits until an actual attempt is made to read or write to the underlying file before raising the exception.

If a runtime security exception is raised in a DASE Application and is not caught by the application, then the application shall be aborted.

If a privileged operation is denied in a context where a runtime security exception is not available, then the DASE System shall not perform the operation.

4.1.1 Effective Permissions Determination

A DASE Application that requires access to a privileged operation listed in ANNEX B shall request permission to perform the operation. Permission to perform an operation shall be requested by means of a permission request resource that adheres to the `application/dase-permission` content type defined by Section 5.1.1.

> *Note:* See [DASE], Section 6.1.1.6.9, for information on how a DASE Application indicates the application entity that represents its permission request resource; in particular, the DASE Application specifies an *entity* element in the application metadata resource with an *entitytype* attribute whose value is `permissionRequest`.

Let *N* be the number of requests in the permission authorization resource and let R_i define the permission request *i*. The algorithm that grants access to privileged operations shall follow the procedure described by the pseudo-code sequence listed below:

```
for i = 1, 2, …,  N
{
    get permission request Ri
    if ( Ri is not denied by the application emission policy )
    {
        if ( Ri is not denied by the local policy )
        {
            system grants permission Ri to application
        }
    }
}
```

Let P represent the set of permissions granted by default to all DASE Applications; that is, P is the set of non-explicitly defined permissions for the unprivileged operations which may be performed by a DASE System. Let G represent the set of granted permissions determined by the algorithm described above. Then, the set of effective permissions, E, is defined as the union of the two sets, that is

$$E = P \cup G$$

Note: This specification does not prescribe the means by which an application emission policy is specified. See [DASE-ARM] for additional information regarding application emission policy for DASE Applications.

Note: This specification does not prescribe the means by which a local policy is specified.

Note: The `java.security.Policy` class may be utilized to manage application emission and local permission policies. See [DASE-PA] Sections 5.1.1.2.1, A.9, and A.10 for information regarding Java security functionality.

Note: The set of non-explicitly defined permissions *P* may be constrained by other policies not specified here.

4.2 Runtime Code Extension

Certain mechanisms of the declarative application environment and procedural application environment permit the augmentation of an application's code (i.e., executable statements or functions) during runtime. Such augmentation is referred to as a *runtime code extension* (RCE).

4.2.1 RCE in Declarative Application Environment

In a declarative application environment, runtime code extension is defined as the performance of any of the following functions by means of the script content facility:

(1) `Global.eval();`
(2) `Function.[[constructor]];`
(3) `HTMLDocument.write() and writeln();`
(4) `Window.setTimeout();`
(5) any function or property which permits the creation or mutation of an intrinsic event attribute;
(6) any function or property which permits the creation or mutation of a *script* element;

When any of the above mechanisms are employed by a DASE application, then, if permission has not been granted to perform a *RuntimeCodeExtension* privileged operation, an ECMAScript runtime security exception shall be raised.

In addition, the declarative application environment shall not permit the removal or replacement of a pre-defined property or function of a host object.

4.2.2 RCE in Procedural Application Environment

In a procedural application environment, runtime code extension is defined as the performance of any of the following functions by means of the active object content facility:

(1) the instantiation of a user defined type which is a subclass of `java.lang.ClassLoader`;

When the above mechanism is employed in the context of any DASE Application, a `java.lang.SecurityException` shall be raised.

5. FACILITIES

This specification defines a facility which specifies a category of content types by enumerating a set of one or more specific content types. The following category is defined:

- permission authorization content

5.1 Permission Authorization Content

This facility consists of a permission authorization content type. Permission authorization content is used to request permission to perform privileged operations.

5.1.1 application/dase-permission

Permission authorization content shall adhere to the *Extensible Markup Language, Version 1.0* [XML] and shall be identified as content type `application/dase-permission`.

Furthermore, permission authorization content shall adhere to the following document type, labeled here according to its formal public identifier, and defined in Section A.1 Permission Authorization Document Type according to procedures set forth in [XML] and [XMLNAMES]:

- `"-//ATSC//DTD DASE Permission 1.0//EN"`

5.1.1.1 Well Formedness

An application entity which employs this content type shall be well formed as prescribed by [XML], Section 2.1.

If an entity of an application uses content type `application/dase-permission`, is not well formed, and the entity is processed, then the application environment shall ignore the entity.

5.1.1.2 Validity

An application entity which employs this content type shall be valid as prescribed by [XML], Section 2.8.

If an entity of an application uses content type `application/dase-permission`, is not valid, and the entity is processed, then the application environment shall ignore the entity.

5.1.1.3 XML Declaration

An application entity which employs this content type shall specify a valid XML declaration [XML:23].

Note: The notation [XML:23] refers to [XML] grammar production 23.

If an entity of an application uses content type `application/dase-permission`, does not specify a valid XML declaration, and the entity is processed, then the application environment shall ignore the entity.

An application entity which employs this content type shall specify an XML declaration with an encoding declaration [XML:80] according to one of the following character encoding systems, and, furthermore, the entity's representation shall employ the specified encoding system as its actual character encoding system.

- `"UTF-8"`
- `"ISO-8859-1"`

If an entity of an application uses content type `application/dase-permission`, does not specify one of the preceding encoding declarations in the XML declaration according to the actual employed character encoding system, and the entity is processed, then the application environment shall ignore the entity.

An application entity which employs this content type shall not specify an XML declaration with a standalone document declaration [XML:32] with the value `"yes"`.

If an entity of an application uses content type `application/dase-permission`, does specify a standalone document declaration with the value `"yes"`, and the entity is processed, then the application environment shall ignore the entity.

5.1.1.4 Document Type Declaration

An application entity which employs this content type shall specify a valid document type declaration [XML:28].

If an entity of an application uses content type `application/dase-permission`, does not specify a valid document type declaration, and the entity is processed, then the application environment shall ignore the entity.

An application entity which employs this content type shall specify a document type declaration with an external identifier [XML:75] containing one of the following public identifiers [XML:12]:

- `"-//ATSC//DTD DASE Permission 1.0//EN"`

If an entity of an application uses content type `application/dase-permission`, does not specify one of the preceding public identifiers in an external identifier in the document type declaration, and the entity is processed, then the application environment shall ignore the entity.

An application entity which employs this content type shall not specify an internal declaration subset. An internal declaration subset is that part of [XML:28] consisting of the delimiters `'['`, `']'`, and all intervening delimiters and non-terminals.

If an entity of an application uses content type `application/dase-permission`, does specify an internal declaration subset (whether empty or not), and the entity is processed, then the application environment shall ignore the entity.

5.1.1.5 Attribute Semantics

Attribute semantics are specified under each element's semantics. See Section 5.1.1.6.

5.1.1.6 Element Semantics

This section specifies the usage and processing semantics of the elements defined for use with this content type.

5.1.1.6.1 *param* element

A *request* element may have one or more optional *param* child elements according to the permission authorizations defined in ANNEX C. If a *request* element specifies a *param* child element whose *name* or *value* is not defined for use with the specific named authorization according to ANNEX C, then the *param* element shall be ignored.

The value of the *name* attribute shall be treated as case-insensitive for the purpose of determining value equality.

Note: No standardized use of the *param* element is defined by this specification; this element is expected to be used in future levels of the DASE Standard.

5.1.1.6.2 *permission* element

The *permission* element serves to aggregate a possibly empty collection of permission authorization specifications. This element does not admit any attribute.

5.1.1.6.3 *request* element

The *request* element specifies a distinct request to be granted permission to perform some operation or some set of operations. The operation(s) requested are determined according to the attributes and children *param* elements of this element type.

A *request* element shall specify a non-empty *name* attribute from the collection of permission authorizations defined in ANNEX C. If a *request* element specifies a value for a *name* attribute not defined in ANNEX C, then the *request* element shall be ignored.

The value of the *name* attribute shall be treated as case-insensitive for the purpose of determining value equality.

A *request* element may specify an optional *target* attribute according to the permission authorizations defined in ANNEX C. If a *request* element specifies a value for a *target* attribute and that value is not permitted for the specific named request according to ANNEX C, then the *request* element shall be ignored.

A *request* element may specify an optional *actions* attribute according to the permission authorizations defined in ANNEX C. If a *request* element specifies a value for an *actions* attribute and that value is not permitted for the specific named request according to ANNEX C, then the *request* element shall be ignored.

ANNEX A. DOCUMENT TYPE DEFINITIONS

The entirety of this section and its subsections is normative.

This annex specifies a document type definition for use with permission authorization content.

A.1 Permission Authorization Document Type

This document type supports permission authorization content.

A.1.1 DTD

```
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                        DASE Permission 1.0 DTD                      -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!--
     This is DASE Permission 1.0, an XML Document Type specified for use with
     ATSC DASE Applications.

     This module shall be idenitifed by the following formal public identifier:

     "-//ATSC//DTD DASE Permission 1.0//EN"
-->

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                              Parameters                             -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!ENTITY % XMLNS "http://www.atsc.org/dase-1#permission" >

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                     Data Type Entity Declarations                   -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!-- a Uniform Resource Identifier, see [URI] -->
<!ENTITY % URI.datatype "CDATA" >

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                     Attribute Entity Declarations                   -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!ENTITY % xmlns.attrib
     "xmlns         %URI.datatype;          #FIXED '%XMLNS;'"
>

<!ENTITY % auth.name.attrib
     "name          CDATA                   #REQUIRED"
>

<!ENTITY % auth.target.attrib
     "target        CDATA                   #IMPLIED"
>

<!ENTITY % auth.actions.attrib
     "actions       CDATA                   #IMPLIED"
>

<!ENTITY % param.name.attrib
     "name          CDATA                   #REQUIRED"
```

```
>

<!ENTITY % param.value.attrib
     "value         CDATA                  #IMPLIED"
>

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--               Qualified Element Name Entity Declarations               -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->

<!ENTITY % permission.qname                "permission" >
<!ENTITY % request.qname                   "request" >
<!ENTITY % param.qname                     "param" >

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                          Element Declarations                         -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->




<!ENTITY % permission.content     "(%request.qname;)+"  >
<!ELEMENT %permission.qname;      %permission.content; >
<!ATTLIST %permission.qname;
      %xmlns.attrib;
>

<!ENTITY % request.content        "(%param.qname;)*" >
<!ELEMENT %request.qname;         %request.content; >
<!ATTLIST %request.qname;
      %auth.name.attrib;
      %auth.target.attrib;
      %auth.actions.attrib;
>


<!ENTITY % param.content          "EMPTY">
<!ELEMENT %param.qname;           %param.content; >
<!ATTLIST %param.qname;
      %param.name.attrib;
      %param.value.attrib;
>

<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
<!--                             END END END                               -->
<!-- ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::: -->
```

ANNEX B. PRIVILEGED OPERATIONS

The entirety of this section and its subsections is normative.

This annex specifies a set of privileged operations, either directly or indirectly by reference to either a declarative application environment operation or a Java *permission* class, each of which either (1) shall always be denied or (2) may be conditionally granted according to whether permission to perform the operation was requested by the application and whether the effective permissions authorize the requested permission.

If an operation is designated as *grantable*, then it may be granted to an application which requests permission to perform the operation. If an operation is designated as *denied*, then it shall never be granted to an application.

Only those operations which have a grantable designation are associated with a *request name*. This request name is used to request permission to perform the operation.

Note: See ANNEX C for further information on permission authorizations.

In the case of an operation associated with the procedural application environment, the operation is designated indirectly by specifying a Java *permission class* which is used to control the performance of the operation.

Note: Further information about the specific operations controlled by a Java permission class can be found in the API specifications referenced by [DASE-PA].

B.1 Declarative Application Environment

This section specifies privilege operations which are specific to the declarative application environment.

Table 1 Declarative Application Environment Privileged Operations

Operation or Feature	Designation	Request Name
Cookie Create, Delete, or Modify	grantable	Cookie
ECMAScript URI Scheme Evaluation	grantable	RuntimeCodeExtension
Event Attribute Create or Modify	grantable	RuntimeCodeExtension
Function.[[constructor]]	grantable	RuntimeCodeExtension
Global.eval	grantable	RuntimeCodeExtension
HTMLDocument.write	grantable	RuntimeCodeExtension
HTMLDocument.writeln	grantable	RuntimeCodeExtension
Script Element Create or Modify	grantable	RuntimeCodeExtension
Service Selection	grantable	Select
Window.setTimeout	grantable	RuntimeCodeExtension
Xlet Instantiation	grantable	Xlet

B.2 Procedural Application Environment

This section specifies privilege operations which are specific to the procedural application environment.

Table 2 Procedural Application Environment Privileged Operations

Permission Class	Designation	Request Name
java.awt.AWTPermission	denied	*none*
java.io.FilePermission	grantable	File
java.io.SerializablePermission	denied	*none*

Permission Class	Designation	Request Name
java.lang.ReflectPermission	denied	*none*
java.lang.RuntimePermission	denied	*none*
java.net.SocketPermission	grantable	Socket
java.security.AllPermission	denied	*none*
java.security.SecurityPermission	denied	*none*
java.util.PropertyPermission	grantable	Property
javax.tv.media.MediaSelectPermission	grantable	MediaSelect
javax.tv.service.ReadPermission	grantable	ServiceInfoAccess
javax.tv.service.selection.SelectPermission	grantable	Select
javax.tv.service.selection.ServiceContextPermission	grantable	ServiceContext
org.atsc.management.ManagementPermission	grantable	StateManagement
org.atsc.preferences.PreferencePermission	grantable	Preference
org.atsc.security.AtscAllPermission	denied	*none*
org.atsc.security.HAViPermission	grantable	DisplayConfig
org.atsc.user.UserPermission	grantable	User
org.atsc.xlet.XletPermission	grantable	Xlet

ANNEX C. PERMISSION AUTHORIZATIONS

The entirety of this section and its subsections is normative.

This annex specifies the targets, actions, and optional parameters of each permission authorization which may appear in a *Request* element of an application entity of type `application/dase-permission`.

C.1 Cookie

Request permission to create, remove, retrieve or modify cookies in the declarative application environment.

The target shall be specified as a URI to indicate the cookie's domain and path. A special target value of `"*"` shall be used to specify a request to access any cookie.

The permitted actions are `"create"`, `"delete"`, `"read"`, and `"write"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

Note: See [DASE-DA], Section 5.3.1.2.3.3.4, for additional information on the prevention of unexpected sharing of cookie state information.

C.2 DisplayConfig

Request permission to modify display configuration.

The target shall be one of the following values in as prescribed by `org.atsc.security.HAViPermission`: `"setBackgroundConfiguration"`, `"setGraphicsConfiguration"`, `"setVideoConfiguration"`, or `"setCoherentScreenConfigurations"`.

No action applies.

Note: See [DASE-API], Section 4.13.4, for additional information on `org.atsc.security.HAViPermission`.

C.3 File

Request permission to access the local file system.

The permitted target is specified as a pathname. A pathname that ends in the special sequence `"/*"` denotes all files and directories in the specified directory. A pathname that ends in the special sequence `"/-"` denotes, recursively, all files and directories in the specified directory. A pathname consisting of the special sequence `"<<ALL FILES>>"` matches any file.

The target values `"*"` and `"-"` shall be interpreted as `"/*"` and `"/-"`, respectively.

The permitted actions are `"read"`, `"write"`, and `"delete"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

An application shall always be granted read access to directories and files which encompass its own automatically mounted application delivery file system. An application need not request such access.

Note: The `write` action applies to file creation as well as to writing an existing file.

Note: For additional information, see [PJAE], `java.io.FilePermission`.

C.4 MediaSelect

Request permission to perform media selection with a Java TV `MediaSelectControl`.

The target shall be specified as a `"tv:"` URI to indicate one or more media streams in a particular service. A special target value of `"*"` shall be used to specify a request to access any media component.

No action applies.

Note: For additional information, see [JAVATV], `javax.tv.media.MediaSelect-Permission`.

C.5 Preference

Request permission to access a preference.

The permitted target is either the special value `"*"`, which denotes all preferences, or a specific preference name.

The permitted actions are `"create"`, `"delete"`, `"read"`, and `"write"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

Note: For additional information, see [DASE-API], `org.atsc.preferences.-PreferencePermission`.

C.6 Property

Request permission to access a system property.

The permitted target is specified as a hierarchical property name. A property name that ends in the special sequence `".*"` shall be interpreted as a wildcard. The target value `"*"` shall denote all properties.

The permitted action is `"read"`. A system shall deny a request that includes an action value other than the one specified here.

Note: For additional information, see [PJAE], `java.util.PropertyPermission.`, and [DASE-PA], Annex C, Java System Properties.

C.7 RuntimeCodeExtension

Request permission to perform a runtime code extension within the declarative application environment as described above in Section 4.2.1.

No target or action applies.

C.8 Select

Request permission to perform service selection.

The target shall be defined as the `"tv:"` URI that identifies the requested service. A special target value of "`*`" shall be used to specify a request to select any service.

The permitted actions are `"*"` or `"own"`. The action `"*"` indicates a request to select a service in any service context, whereas the action `"own"` indicates a request to select a service in the current service context.

Note: For additional information, see [JAVATV], `javax.tv.service.selection.-SelectPermission`.

C.9 ServiceContext

Request permission to perform certain operations on a service context.

The permitted target values are `"access"` or `"getServiceContentHandlers"`. A system shall deny a request that includes a target value other than the ones specified here.

The permitted actions are `"*"` or `"own"`. The action `"*"` indicates a request to perform the operation on a service in any service context, whereas the action `"own"` indicates a request to perform the operation on a service in the current service context.

Note: For additional information, see [JAVATV], `javax.tv.service.selection.-ServiceContextPermission`.

C.10 ServiceInfoAccess

Request permission to access service information.

The target shall be a `"tv:"` URI that identifies a request to access service information for a particular virtual channel. A special target value of `"*"` shall be used to specify a request to access service information for any virtual channel.

No action applies.

Note: For additional information, see [JAVATV], `javax.tv.service.Read-Permission`.

C.11 Socket

Request permission to access a datagram socket.

The permitted target is specified as a `socketTarget` specification as follows:

```
socketTarget := host [ ":" portRange ]
host         := "localhost" | ipV4Address
ipV4Address  := 1*digit "." 1*digit "." 1*digit "." 1*digit
portRange    := port | "-" port | port "-" port
port         := 1*digit
```

A `port` specification of *N* means any port number greater than or equal to *N*. A `port` specification of *-N* means any port number less than or equal to *N*. A `port` specification of *M-N* means any port number from *M* to *N*, inclusive.

The permitted actions are `"accept"`, `"connect"`, and `"listen"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

Note: For additional information, see [PJAE], `java.net.SocketPermission`.

C.12 StateManagement

Request permission to modify a resource's state or status management attributes.

The permitted target values are `"lock"`and `"clear"`.

No action applies.

Note: For additional information, see [DASE-API], `org.atsc.user.Management-Permission`.

C.13 User

Request permission to access user capabilities in the procedural application environment.

The permitted target is one of the following: (1) the special value `"user"`, which is used to perform operations on the user registry, (2) the special value `"*"`, which is used to perform operations on individual user capabilities and denotes all user capabilities, or (3) a specific user capability name, which is used to perform operations on individual user capabilities.

When used to perform operations on the user registry, the permitted actions are `"create"`, `"delete"`, `"read"`, and `"write"`. When used to perform operations on individual user capabilities, the permitted actions are `"confer"` and `"retract"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

Note: For additional information, see [DASE-API], `org.atsc.user.UserPermission`.

C.14 Xlet

Request permission to (1) embed an Xlet in a declarative application, (2) register or deregister an Xlet in the Xlet registry, or (3) access or control the state of an Xlet in the current DASE Application.

The permitted target is specified as a resource identifier which denotes the Xlet's class resource or the special target `"*"`.

The permitted actions are `"embed"`, `"get"`, `"pause"`, `"register"`, `"resume"`, `"start"`, `"stop"`, and `"unregister"`. Multiple actions may be specified in a comma-separated list with optional intervening whitespace.

The action `"embed"` shall be used by a declarative application environment to control the ability of a declarative application to load and activate an *embedded* Xlet. If permission to embed an Xlet is not granted to a declarative application, the application shall not be aborted, but shall be allowed to operate without processing any embedded Xlet.

Note: For additional information, see [DASE-API], `org.atsc.xlet.XletPermission`.

ANNEX D. EXAMPLES

The entirety of this section and its subsections is informative.

This annex presents examples of the use of the content types defined by this specification.

D.1 Permission Authorization Request Example

This example depicts a permission authorization request for a DASE Application. It contains requests for four types of privileged operations: (1) embed an Xlet in a declarative application, (2) perform ECMAScript related runtime code extension techniques, (2) perform file system access, and (4) perform service selection.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE permission PUBLIC "-//ATSC//DTD DASE Permission 1.0//EN">
<permission>
  <!-- request permission to embed an Xlet in a declarative app. -->
  <request name="Xlet" target="*" actions="embed"/>
  <!-- request permission to use run time code extensions -->
  <request name="RuntimeCodeExtension"/>
  <!-- request permission to access a particular local file -->
  <request name="File" target="/com/tv/info.dat" actions="read,write"/>
  <!-- request permission to use the service selection java class -->
  <request name="Select" target="*" actions="*"/>
</permission>
```

CHANGES

This section is informative.

Changes from Candidate Standard to Proposed Standard

The following table enumerates the changes between the issuance of the candidate standard edition of this specification and the proposed standard edition.

Table 3 Changes from Candidate Standard

Section	Description
1	Change status to approved proposed standard.
B.1	Designate ECMAScript URI Scheme Evaluation as grantable via RuntimeCodeExtension.
B.2	Always deny java.io.SerializablePermission.
C	Change name of SelectContext permission to ServiceContext.
C.3	Always grant read access to automatically mounted application delivery file system entities.
C.6	Restrict actions on PropertyPermission to "read" only.
C.9	Change name of SelectContext permission to ServiceContext.
C.9	Correct reference to read javax.tv.service.selection.ServiceContextPermission.