

Unabhängige Treuhandstelle

UNIVERSITÄTSMEDIZIN GREIFSWALD

Stand: Juli 2026

Rollenbasierte Domänenabsicherung

Ziel

Mit **rollenbasierter Domänenabsicherung** können einzelne Domänen der Tools **für authentifizierte Benutzer**, basierend auf den ihnen zugeordneten Rollen, ein- bzw. ausgeblendet werden. So werden über spezielle Rollen *die* Domänen beschrieben, auf die der Zugriff erlaubt sein soll. Alle anderen Domänen werden "ausgeblendet".

Als **Paradigma** wird dabei die **transparente "Perspektive"** (oder "View") verwendet: Anfragen zur Liste aller Domänen werden nur mit den zugelassenen Domänen beantwortet und Zugriffsversuche auf andere Domänen werden so beantwortet, als gäbe es diese gar nicht. (Konkret werden solche Anfragen mit `UnknownObjectException` pariert statt mit `AuthenticationException` o.ä.) So ist es einem Nutzer auch nicht möglich, durch gezielte Anfragen herauszufinden, welche weiteren Domänen in der Instanz vorhanden sind.

Die **"Filterung"** der Domänen erfolgt **im Backend**, so dass die Zugriffe über **alle Schnittstellen entsprechend eingeschränkt** werden, sofern sie authentifiziert und mit aktivierter rollenbasierter Domänenabsicherung erfolgen.

Das zweistufige Rollensystem mit `admin`- und `user`-Rollen bleibt von rollenbasierter Domänenabsicherung unberührt und kann als komplementär dazu betrachtet werden.

Aktivierung

Um rollenbasierte Domänenabsicherung zu nutzen, müssen die Schnittstellen der Tools grundsätzlich abgesichert verwendet werden: es muss also die Authentifizierung der einzelnen Schnittstellen aktiviert sein. Das funktioniert für alle Schnittstellen (wie gehabt) z.B. über `.env`-Variablen in der Docker-Konfiguration:

- für **Web per OIDC** (über Elytron-OIDC aktivierbar) in `envs/ttp_TOOL.env` mit
  - `TTP_EPIX_WEB_AUTH_MODE=keycloak`
  - `TTP_GICS_WEB_AUTH_MODE=keycloak`
  - `TTP_GPAS_WEB_AUTH_MODE=keycloak`
- für **Web per gRAS** (über Elytron-JDBC) in `envs/ttp_TOOL.env` mit
  - `TTP_EPIX_WEB_AUTH_MODE=gras`
  - `TTP_GICS_WEB_AUTH_MODE=gras`
  - `TTP_GPAS_WEB_AUTH_MODE=gras`

- für **SOAP per OIDC** (über KeyCloak-API) in `envs/ttp_TOOL.env` mit
  - `TTP_EPIX_SOAP_KEYCLOAK_ENABLE=true`
  - `TTP_GICS_SOAP_KEYCLOAK_ENABLE=true`
  - `TTP_GPAS_SOAP_KEYCLOAK_ENABLE=true`
- für **FHIR per OIDC** (über KeyCloak-API) in `envs/ttp_common.env` mit
  - `TTP_FHIR_KEYCLOAK_ENABLE=true`

Um nun zusätzlich die rollenbasierte Domänenabsicherung zu steuern, gibt es drei Varianten:

- `DISABLED`
  Die rollenbasierte Domänenabsicherung ist deaktiviert, d.h. **alle Benutzer** haben Zugriff auf **alle Domänen**.
- `FORCED`
  Die rollenbasierte Domänenabsicherung wird für alle Benutzer erzwungen. D.h. für alle Benutzer ist der Zugriff nur auf *die* Domänen möglich, für die ihnen entsprechende Domänenrollen zugeordnet sind. Insbesondere haben in diesem Modus **Benutzer ohne** domänenbasierte Rollen Zugriff auf **keine Domänen**.
- `IMPLIED`
  Die Aktivierung der rollenbasierten Domänenabsicherung wird aus den Rollen der Benutzer abgeleitet. D.h. wenn Benutzer domänenbasierte Rollen haben, wird für sie die rollenbasierte Domänenabsicherung aktiviert und für sie ist der Zugriff nur auf *die* Domänen möglich, für die ihnen entsprechende Domänenrollen zugeordnet sind. Insbesondere haben in diesem Modus **Benutzer ohne** domänenbasierte Rollen Zugriff auf **alle Domänen**.

Ohne weiteres Zutun ist die rollenbasierte Domänenabsicherung implizit aktiviert, `IMPLIED` ist also der Standardmodus. Um die anderen Modi zu aktivieren, müssen (jeweils einzeln für die Tools) entsprechende `.env`-Variablen in der Docker-Konfiguration (in `envs/ttp_TOOL.env`) gesetzt werden, z.B.:

- `TTP_EPIX_AUTH_DOMAIN_ROLES=FORCED`
- `TTP_GICS_AUTH_DOMAIN_ROLES=FORCED`
- `TTP_GPAS_AUTH_DOMAIN_ROLES=DISABLED`

Alternativ können manuell in der `standalone.xml` des **WildFly** entsprechende Systemproperties eingetragen werden:

```
<system-properties>
  ...
  <property name="ttp.auth.epix.domain.roles" value="FORCED"/>
  <property name="ttp.auth.gics.domain.roles" value="FORCED"/>
  <property name="ttp.auth.gpas.domain.roles" value="DISABLED"/>
  ...
</system-properties>
```

Freischaltung von Domänen durch spezielle Rollen

Um bei aktivierter rollenbasierter Domänenabsicherung für einzelne Benutzer konkrete Domänen freizuschalten, müssen den Benutzern im **AllowList-Prinzip** in der Administration eines **OIDC-Servers** (z.B. Keycloak) oder in der **gRAS-Datenbank** spezielle Rollen zugeordnet werden.

Diese Zuordnung erfolgt per Namenskonvention: eine Domänenrolle ist ein **Muster** (Pattern), das eine **konkrete Domäne per Namen** oder auch eine **ganze Klasse von Domänennamen** beschreiben kann. Als **Mechanismus** werden dabei vereinfachte **Glob-Wildcards** verwendet (ohne Zeichenklassen, Bereiche oder Komplementierung). Das Matching erfolgt immer **case-insensitiv**.

Eine Domänenrolle hat die Form `:TOOL:DOMAIN`, wobei `TOOL` und `DOMAIN` natürlich mit Wildcards ausgedrückt werden können. Formal passt eine Domänenrolle immer mindestens auf die *Regex* `^:[^:]+:[^:]+`). Einige illustrierende Beispiele:

- `:*:*` erlaubt den Zugriff auf alle Domänen im E-PIX und im gICS und die Root-Domänen im gPAS
- `:epix:*` erlaubt den Zugriff auf alle Domänen im E-PIX
- `:epix:demo` erlaubt den Zugriff auf die Domäne "Demo" im E-PIX
- `:*:demo` erlaubt den Zugriff auf die "Demo"-Domänen in allen Tools und die "Demo"-Root-Domänen im gPAS
- `:gics:mii*` erlaubt im gICS den Zugriff auf alle mit "MII" beginnende Domänen
- `:gics:xyz ?? v2.?` erlaubt im gICS z.B. den Zugriff auf "XYZ DE v2.0" und "XYZ EU v2.1" aber nicht "XYZ v2.0" oder "XYZ DE v2"
- `:*:jmeter*` erlaubt den Zugriff auf alle JMeter-Test-Domänen im E-PIX und im gICS und die JMeter-Root-Domänen im gPAS

Zur Illustration hier noch eine Übersicht für die Kombination der drei Aktivierungsmodi mit speziellen Domänenrollen beispielhaft für den Zugriff auf die Domäne `MII` im gICS:

Modus	**Ohne Domänenrollen**	`:*:*`	`:*:mii`	`:gics:mii`	`:gics:demo`	`:gics:*`	`:epix:mii`
`DISABLED`	YES	YES	YES	YES	YES	YES	YES
`FORCED`	**NO**	YES	YES	YES	**NO**	YES	**NO**
`IMPLIED`	YES	YES	YES	YES	**NO**	YES	**NO**

Domänenpfade im **gPAS**

Im **gPAS** gibt es die Besonderheit, dass ganze Domänenpfade beschrieben werden müssen. Im Gegensatz zu Globbing-Wildcards für Dateipfade wird hier `:` (statt `/`) als Trennzeichen für Einzelteile der Pfade verwendet. Ein einzelner `*` kann nur innerhalb eines Teiles (einer Domäne) matchen (zwischen zwei `:`), ein doppelter `**` aber über Teilgrenzen (Domänen) hinweg.

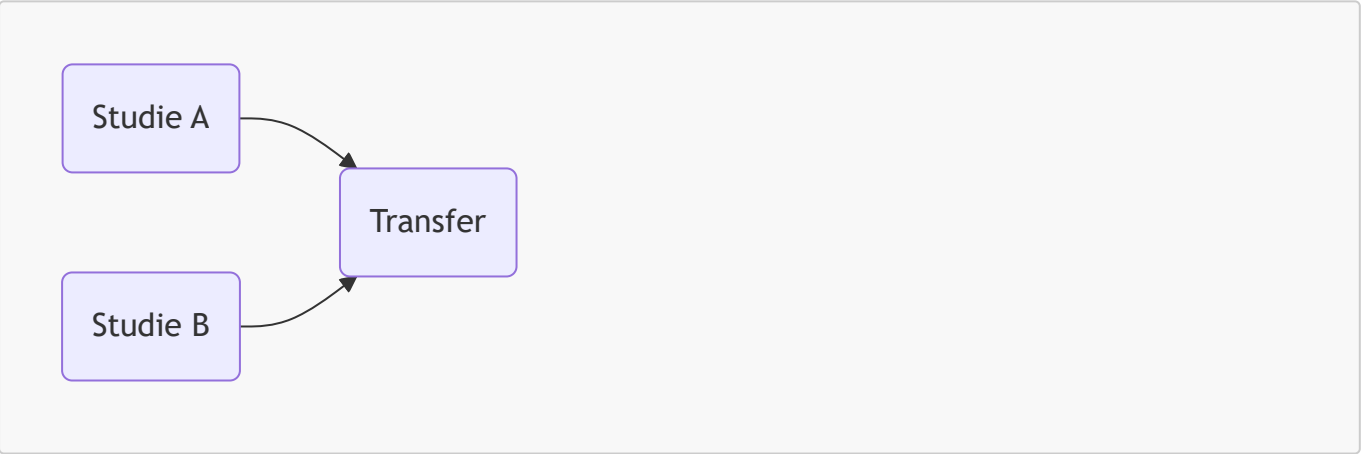
gPAS-Domänenrollen haben die Form `:TOOL:DOMAIN:SUBDOMAIN:SUBSUBDOMAIN:...` und formal passt eine **gPAS**-Domänenrolle mindestens auf die *Regex* `^:[^:]+:[^:]+.*` bzw. `^:gpas:[^:]+.*`, wenn sie nur für den **gPAS** gelten soll.

Ein doppelter Doppelpunkt (`::`) kann zusätzlich als "Anker" verwendet werden und inkludiert dann auch die jeweils passende Domäne vor dem `::`. Die Verwendung des doppelten Doppelpunkts kann als Abkürzung für die Verwendung von zwei Rollen betrachtet werden. Beispielsweise ist die Rolle `:gpas:x::y`gleichwertig zu den beiden Rollen `:gpas:x` und `:gpas:x:y` oder die Rolle `:gpas:**:x::**:y`gleichwertig zu den beiden Rollen `:gpas:**:x` und `:gpas:**:x:**:y` (z.B. passend für `a:x` und `a:x:b:y` aber nicht für `a:x:y`).

Auch hierzu einige illustrierende Beispiele:

- `:*:**` erlaubt den Zugriff auf alle Domänen in allen Tools, insbesondere auch auf alle Domänenpfade im gPAS
- `:g*:**` erlaubt den Zugriff auf alle Domänenpfade im gPAS und alle Domänen im gICS
- `:gpas:studie a:biolabor` erlaubt den Zugriff auf den Domänenpfad "Studie A:Biolabor" im gPAS
- `:gpas:**` erlaubt den Zugriff auf alle Domänenpfade
- `:gpas:studie a:**` erlaubt den Zugriff auf alle Domänenpfade unterhalb von "Studie A" im gPAS, aber nicht "Studie A" selbst
- `:gpas:studie ?::**` erlaubt den Zugriff auf alle Domänenpfade unterhalb von "Studie A", "Studie B", ... im gPAS inklusive dieser selbst
- `:gpas:studie 1::**.system**` erlaubt den Zugriff auf alle Domänenpfade in "Studie 1" und unterhalb von "Studie 1" wenn deren Pfad eine Domäne mit `.system` im Namen enthält
- `:gpas:studie**system` erlaubt den Zugriff auf alle Domänenpfade deren erste Domäne mit "Studie" startet und deren letzte Domäne mit `system` endet
- `:gpas:**:biolabor` erlaubt den Zugriff auf alle Domänenpfade mit Biolaboren am Ende
- `:*:jmeter**` erlaubt den Zugriff auf alle JMeter-Test-Domänen in allen Tools

Zusätzliche Komplexität entsteht im **gPAS** dadurch, dass **Domänen mit mehreren Elterndomänen** verknüpft sein können, z.B.



Hier kann die Domäne `Transfer` beispielsweise durch die Rolle `:gpas:Studie A:Transfer` oder `:gpas:Studie B:Transfer` freigeschaltet werden. In `Transfer` kann es aber konkrete Pseudonyme mit Originalwerten aus `Studie A` **und** aus `Studie B` geben.

Wenn hier `Transfer` über irgendeinen Domänenpfad freigeschaltet ist, werden in der aktuellen gPAS-Version 2026.1.0 immer **ALLE** Pseudonyme aus `Transfer` angezeigt, die mit Originalwerten aus `Studie A` **und** die mit Originalwerten aus `Studie B`, egal, ob die Domäne über `:gpas:Studie A:Transfer` oder `:gpas:Studie B:Transfer` freigeschaltet wurde (*all-or-none*).

Allgemein gilt also:

> Wenn eine Domäne durch eine bestimmte Regel freigeschaltet ist, sind immer **ALLE** Pseudonyme dieser Domäne sichtbar, auch die aus anderen Domänenpfaden, also auch jene, deren konkrete Pseudonympfade der bestimmten Regel nicht entsprechen (*all-or-none*).
>
> Diese scheinbare Diskrepanz kann nur bei Pfaden auftreten, die Domänen mit multiplen Elterndomänen enthalten.

In einer zukünftigen Version soll die Einschränkung der Sichtbarkeit auch für konkrete Pseudonym-Pfade umgesetzt werden.

Fehlersuche

Hilfreich für das Debugging möglicher Probleme könnte der `DEBUG`-Modus verschiedener Logger sein. Für die Docker-Konfiguration kann dieser mit folgenden `.env`-Variablen aktiviert werden:

```
WF_CONSOLE_LOG_LEVEL=DEBUG
TTP_EPIX_LOG_LEVEL=DEBUG
TTP_GICS_LOG_LEVEL=DEBUG
TTP_GPAS_LOG_LEVEL=DEBUG
TTP_FHIR_LOG_LEVEL=DEBUG
TTP_AUTH_LOG_LEVEL=DEBUG
TTP_WEB_LOG_LEVEL=DEBUG
```

Die Variablen sind verteilt auf `ttp-common.env`, `ttp_TOOL.env` und `ttp_fhir.env` in `envs/`.