Break Glass Role Management (-=DRAFT=-) (-=IN PROGRESS=-)

About

Break glass role management (BGRM) is a role management framework where accounts are only provided role rights when needed. Unlike persistent roles where the rights granted remain available at all times, break glass roles are only granted for a limited time.

This guide is to help ServiceNow admins setup BGRM for user roles insuring accounts are properly setup for privileged rights. The goal for BGRM is to work in conjunction with the default out-of-the-box (OOTB) ServiceNow role base security model to insure future upgradeability across ServiceNow releases. Therefore any role based security already setup in ServiceNow will not be impacted.

Disclaimer

There have been many questions around roles and licensing pertaining to BGRM. BGRM is not about reducing licensing costs or circumventing licensing. BGRM is focused on granting ad-hoc elevated security permissions. The standard rule of thumb is when it comes to licensing is always check with your ServiceNow account representative; whatever is documented in this guide is subject to change

as ServiceNow is free to change their licensing model at anytime. There can be custom roles that can have an associated license cost and there can be custom roles that have no additional licensing costs. This all depends on how solutions are configured within the instance.

How BGRM works

BGRM uses the existing OOTB roles and access control lists used in ServiceNow. Refer to the ServiceNow documentation for information on roles and access control lists). The only difference with BGRM is you do not grant an account the role directly like you do with ServiceNow, you utilize the BGRM framework to automatically grant to and revoke from the user's account.

A break glass role record has the following components:

- Name: The name of the break glass role. This can be the same or different from the ServiceNow role associated with the record.
- Description: Helpful description of the break glass role.
- Group: A reference to the *Group [sys_user_table]* table. This group has the specific ServiceNow roles applied to it where the user will added to.
- User Criteria: A reference to the *User Criteria [user_criteria]* table. This determines if the user has access to the break glass role.
- Lifespan: The number of minutes a user can have the break glass role.

After the standard ServiceNow roles and ACLs are created, then apply the roles to the designated groups. Then a break glass role can be created pointing to the Group [sys_user_group] record. This creates a mapping between what roles a user can temporarily have and the users that can use them.

There is a catalog item called **Break Glass Roles** that will allow users to request access to the break glass roles listed in the *Break Glass Role [break_glass_role]* table. Each break glass role is associated with a user criteria, so only the break glass roles that user meets the user criteria with are displayed. This is the reduce overhead of authorization and approval manual steps. Once the request is made, automation provisions the break glass role for the user.

In the background, once an account's break glass role has expired, the user's account will be automatically removed from the group and the account's sessions will be forcefully terminated.

> **MFA with BGRM**
>
> ServiceNow offers multi-factor authentication (MFA) with one-time password (OTP) for role based accounts. The BGRM is built around just in time authorization for security, so it would make sense to consider leveraging MFA OTP on the roles created for BGRM. That way, when a break glass role is granted to an account and the account is forced to re-login, then the account will be presented with a MFA OTP prompt for additional security based on the newly assigned role. Refer to the ServiceNow documentation for setting up Multi-factor authentication.

BGRM versus Elevated Roles

ServiceNow does not have a temporary model for roles; roles are provisioned until manually removed. While it is possible to designate a role as an elevated role, role elevation restriction only remains on the platform navigation of ServiceNow, not on the portals or applications. Also, role elevation lacks the temporal aspect of security privileges.

Follow along

ServiceNow offers many ways to solve a problem or configure an operational business model. These instructions are *a* way and does not represent *the* way on creating a break glass role framework. The best way to use these instructions is to read through them and see how the framework can be adopted and modified to fit your organization requirements.

Update sets

There is an update set that implement this guide; Break Glass Role Management. The update set can be found at https://github.com/ChristopherCarver/BreakGlassRoleMgt.

Modifying OOTB Tables

This guide focuses on creating a robust framework tied closely to the out of the box (OOTB) tables already existing in ServiceNow. There are alternative implementation solutions within ServiceNow to accomplish the same result. Your mileage may vary depending on scope defined by and practices set by your organization and/or development team. This framework is meant to be as open and flexible to meet varying modifications to suit present and future requirements.

Cost Impact

Custom tables are created in this framework and there could be a financial impact creating custom tables. ServiceNow allots a set amount of custom tables within an instance that a customer can create free of charge. After the set amount of custom tables are created, ServiceNow charges for custom tables. Talk with your ServiceNow account representative to learn more.

Reduce Cost Impact

To reduce any cost impact to your organization, you can extend OOTB tables free of charge. This guide leaves it to the implementation team to determine the best course of action.

BGRM Foundation

The foundation for the BGRM framework is based on the premise that the framework is separate from the default OOTB ServiceNow security model. The only overlap between the BGRM framework and standard ServiceNow user role model is the BGRM relies on ServiceNow to enforce role based security once an account has been granted a role by the BGRM framework. The BGRM framework takes care of the management of the roles independently.

Break Glass Role Table

The *Break Glass Role [u_break_glass_role]* table contains all of the break glass roles available. This table will need to be manually populated by the admin.

The *Break Glass Role [u_break_glass_role]* table will contain the following custom fields:

Field	Description
Name	The name of the break glass role.
Description	A description of the break glass role.
Group	Reference to the ServiceNow table *Group [sys_user_group]*.
User Criteria	Reference to the ServiceNow table *User Criteria [user_criteria]*.
Lifespan	The number of minutes an account can be granted the role; default 4 hours.

Instructions:

1. Navigate to **System Definition > Tables**.
2. Click **New**.
3. Under the **Table New record** section, fill in the following fields:
    - Label: Break Glass Role
    - Name: u_break_glass_role
    - Add module to menu: [User Administration]
4. In the record header, right-click and select **Save**.
5. In the **Columns** table, click **New**.
6. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [String]
    - Column label: Name
    - Column name: {*this should default to u_name*}
    - Max length: 16
    - Mandatory: [true]
7. Click **Submit**.
8. In the **Columns** table, click **New**.
9. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [String]

- Column label: Description
- Column name: {*this should default to u_description*}
- Max length: 64

10. Click **Submit**.
11. In the **Columns** table, click **New**.
12. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Reference]
    - Column label: Group
    - Column name: {*this should default to u_role*}
    - Mandatory: [true]
13. In the **Reference Specification** tab, fill in the following field:
    - Reference: Group [sys_user_group]
14. Click **Submit**.
15. In the **Columns** table, click **New**.
16. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Reference]
    - Column label: User Criteria
    - Column name: {*this should default to u_user_criteria*}
    - Mandatory: [true]
17. In the **Reference Specification** tab, fill in the following field:
    - Reference: User Criteria [user_criteria]
18. Click **Submit**.
19. In the **Columns** table, click **New**.
20. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Integer]
    - Column label: Lifespan
    - Column name: {*this should default to u_lifespan*}
21. In the **Default Value** tab, fill in the following field:
    - Default Value: 240
22. Click **Submit**.
23. Click **Update**.

Break Glass Role Slush Bucket

The *Break Glass Role [u_break_glass_role]* table is referenced by the *Break Glass Roles* catalog item and the reference fields need to defined in a slush bucket for selection.

1. In the browser URL type in **https://{your instance}/u_break_glass_role_list.do?sysparm_view=sys_ref_list**, where {your instance} is the name of your ServiceNow instance.
2. Right-click in the table column header and select **Configure > List Layout**.
3. In under the **Available** list add the following columns to the **Selected** list by selecting the name of the column and click **>**.
    - Name
    - Description

4. If there are any other fields under in the **Selected** list, remove them by selecting the field name and clicking **<** to remove from the **Selected** list.
5. Click **Save**.

Break Glass User Role Table

The *Break Glass User Role [u_break_glass_user_has_role]* table contains the relationship between the account and the break glass role. This table should be maintained by the catalog item *Break Glass Roles* and the supporting flow.

The *Break Glass User Role [u_break_glass_user_has_role]* table will contain the following custom fields:

Field	Description
User	Reference to the ServiceNow OOTB table *User [sys_user]*.
Role	Reference to the custom table *Break Glass Role [u_break_glass_role]*.
Active	Designation that the account has been granted the role.
Expiration	When the role is set to expire on the account.
Request item	Reference to the ServiceNow table *Requested Item [sc_req_item]*.

Instructions:

1. Navigate to **System Definition > Tables**.
2. Click **New**.
3. Under the **Table New record** section, fill in the following fields:
    - Label: Break Glass User Role
    - Name: u_break_glass_user_has_role
    - Add module to menu: [User Administration]
4. In the record header, right-click and select **Save**.
5. In the **Columns** table, click **New**.
6. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Reference]
    - Column label: User
    - Column name: {*this should default to u_user*}
    - Mandatory: [true]
7. In the **Reference Specification** tab, fill in the following field:
    - Reference: User [sys_user]
8. Click **Submit**.
9. In the **Columns** table, click **New**.
10. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Reference]
    - Column label: Break Glass Role
    - Column name: {*this should default to u_break_glass_role*}
    - Mandatory: [true]
11. In the **Reference Specification** tab, fill in the following field:

- Reference: Role [u_break_glass_role]

12. Click **Submit**.
13. In the **Columns** table, click **New**.
14. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [True/False]
    - Column label: Active
    - Column name: {*this should default to u_active*}
15. In the **Default Value** tab, fill in the following field:
    - Default Value: true
16. Click **Submit**.
17. In the **Columns** table, click **New**.
18. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Date/Time]
    - Column label: Expiration
    - Column name: {*this should default to u_expiration*}
    - Mandatory: [true]
19. Click **Submit**.
20. In the **Columns** table, click **New**.
21. In the **Dictionary Entry New record** section, fill in the following fields:
    - Type: [Reference]
    - Column label: Request item
    - Column name: {*this should default to u_request_item*}
    - Mandatory: [true]
22. In the **Reference Specification** tab, fill in the following field:
    - Reference: Requested Item [sc_req_item].
23. Click **Submit**.
24. Click **Update**.

Break Glass Util

The BreakGlassUtil script include supports various break glass operations used in break glass catalog items and flows.

Instructions:

1. Navigate to **System Definition > Script Includes**.
2. Click **New**.
3. In the **Script Include New record** section, fill in the following fields:
    - Name: BreakGlassUtil
    - API Name: {*this should default to global.BreakGlassUtil*}
    - Client callable: [false]
    - Description: Utility operations in the support for break glass roles. See *Break Glass Role [u_break_glass_role]* and *Break Glass User Role [u_break_glass_user_has_role]* tables. And the *Break Glass Roles* catalog item.
    - Script:

```
var BreakGlassUtil = Class.create();
BreakGlassUtil.prototype = {
    initialize: function() {},

    // returns a comma delimited string of break glass roles the user has access t
    getRoles: function() {
        var breakGlassRoleList = [];

        // get all the break glass roles
        var breakGlassRole = new GlideRecord('u_break_glass_role');
        breakGlassRole.addEncodedQuery("u_group.active=true");
        breakGlassRole.query();
        while (breakGlassRole.next()) {
            var criteria = [];
            criteria.push(breakGlassRole.getValue('u_user_criteria'));
            // if the user matches the user criteria for the role
            if (sn_uc.UserCriteriaLoader.userMatches(gs.getUserID(), criteria)) {
                // then store it in the array
                breakGlassRoleList.push(breakGlassRole.getValue('sys_id'));
            }
        }

        // return the list of break glass roles the user has access to as a comma
        return breakGlassRoleList.join(',');
    },
    type: 'BreakGlassUtil'
};
```

4. Click **Submit**.

Break Glass Roles

The Break Glass Roles catalog item is a self-service account elevation role provisioning service.

Instructions:

1. Navigate to **Service Catalog > Catalog Definitions > Maintain Items**.
2. Click **New**.
3. In the **Catalog item New record** section, fill in the following fields:
    - Name: Break Glass Roles
    - Catalogs: [Service Catalog] {*choose the right catalog for you*}
    - Category: [Role Delegation] {*choose the right category for you*}
    - Fulfillment automation level: [Fully automated]
4. In the record header, right-click and select **Save**.
5. In the **Item Details** tab, fill in the following fields:
    - Short Description: Temporary account role elevation.

- Description:

  About: The *Break Glass Roles* catalog item temporary grants your account an elevated break glass roles. A break glass role is a temporary privileged role that an account checks out to perform elevated operations.

  Usage: Only the break glass roles that are available to your account will appear in the *Role* field.

  Important:

  - This is a fully automated request. After your submit your request, your ServiceNow session will be forcible terminated by ServiceNow and you will have to log back into ServiceNow. Upon logging into ServiceNow your account will have the requested role temporarily.
  - Once your temporary break glass role expires, the role will automatically be removed from your account and your ServiceNow session will be forcible terminated by ServiceNow.
  - If your ServiceNow session is not terminated, then check your request to determine the reason why your request was not granted.

6. In the record header, right-click and select **Save**.
7. In the **Variables** tab, click **New**.
8. In the **Variable New record** section, fill in the following fields:
   - Type: [Single Line Text]
   - Mandatory: [true]
   - Order: 100
9. In the **Question** tab, fill in the following fields:
   - Question: Justification
   - Name: {*should default to justification*}
10. Click **Submit**.
11. In the **Variables** tab, click **New**.
12. In the **Variable New record** section, fill in the following fields:
    - Type: [List Collector]
    - Mandatory: [true]
    - Order: 200
13. In the **Question** tab, fill in the following fields:
    - Question: Roles
    - Name: {*should default to roles*}
14. In the **Type Specifications** tab, fill in the following fields:
    - List table: Break Glass Role [u_break_glass_role]
    - Reference qualifier:

```
javascript: var query;
query = "sys_idIN" + new BreakGlassUtil().getRoles();
query;
```

- Variable attributes:

```
no_filter,ref_auto_completer=AJAXTableCompleter,ref_ac_columns=u_name;u_descriptic
```

15. Click **Submit**.
16. Click **Update**.

Break Glass Role Process Engine

The Break Glass Roles catalog item will have a custom process engine to process the request.

Diagram:

Here is the calling hierarchy outline of the custom operations starting from the catalog item *Break Glass Roles* calling the flow, subflows, and actions.

graph LR; A[Break Glass Roles] B{{Break Glass Roles - Catalog Item}} C{{Break Glass Grant Role}} D{{Break Glass Close User Sessions}} E{{Break Glass Get Expirations}} F{{Break Glass Grant Role Banner}} A-->B; B-->C; B-->D; C-->E; C-->F;
To prevent having to go back and forth when creating parent and child components in the documentation, this guide will build out the process engine flow backwards in this order:

1. Break Glass Get Expirations
2. Break Glass Grant Role Banner
3. Break Glass Grant Role
4. Break Glass Close User Sessions
5. Break Glass Roles - Catalog Item

This will allow the parent to link to the existing child operation component, without having to go back and update parent calling operations.

Break Glass Get Expirations

The *Break Glass Get Expirations* is an action component within the flow operations. The purpose of the action is to determine when the break glass role is expiring and expires. Both of these dates and times are used for banners and used to determine when to revoke the break glass role from the user's account.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Action**.
3. Fill in the following fields: -Action name: Break Glass Get Expirations -Description: Calculate the break glass role expiring and expires date and time based on the current time and the lifespan of the break glass role.

4. Click **Submit**.
5. Under **Action Outline**, click **Inputs**.
6. In the **Action Inputs** section, click **Create Input** and fill in the following fields:
    - Label: Break Glass Role
    - Name: break_glass_role
    - Type: (**Reference > Break Glass Role**)
    - Mandatory: [true]
7. Under **Action Outline**, between **Inputs** and **Error Evaluation**, click the **+** sign.
8. In the **Choose a step to add to your action**, find and select **Script**.
9. Under **Script Step**, under **Input Variables**, click **Create Variable**, and fill in the following fields:
    - Name: lifespan
    - Value: (**Inputs > Break Glass Role > Lifespan**)
10. Under **Script** add the following:

```
(function execute(inputs, outputs) {
  var lifespan = inputs.lifespan;

  // calc. expiring
  var lifespanExpiring = parseInt(lifespan * 0.9, 10); // 90% of the lifespan in minutes
  var nowExpiring = new GlideDateTime();
  nowExpiring.addSeconds(lifespanExpiring * 60);
  outputs.expiring_system = nowExpiring.getValue();
  outputs.expiring_local = nowExpiring.getDisplayValue();

  // calc. expires
  var nowExpires = new GlideDateTime();
  nowExpires.addSeconds(lifespan * 60);
  outputs.expires_system = nowExpires.getValue();
  outputs.expires_local = nowExpires.getDisplayValue();

})(inputs, outputs);
```

11. Under **Output Variables**, click **Create Variable** and fill in the following fields:
    - Label: Expiring System
    - Name: expiring_system
    - Type: [Date/Time]
    - Mandatory: [true]
    1. Under **Output Variables**, click **Create Variable** and fill in the following fields:
    - Label: Expiring Local
    - Name: expiring_local
    - Type: [Date/Time]
    - Mandatory: [true]
12. Under **Output Variables**, click **Create Variable** and fill in the following fields:
    - Label: Expires System
    - Name: expires_system
    - Type: [Date/Time]
    - Mandatory: [true]
13. Under **Output Variables**, click **Create Variable** and fill in the following fields:
    - Label: Expires Local

- Name: expires_local
- Type: [Date/Time]
- Mandatory: [true]

14. Under **Action Outline**, click **Outputs**.
15. Under **Action Output**, click **Edit Outputs**.
16. Click **Create Output** and fill in the following fields:
    - Label: Expiring System
    - Name: expiring_system
    - Type: [Date/Time]
    - Mandatory: [true]
17. Click **Create Output** and fill in the following fields:
    - Label: Expiring Local
    - Name: expiring_local
    - Type: [Date/Time]
    - Mandatory: [true]
18. Click **Create Output** and fill in the following fields:
    - Label: Expires System
    - Name: expires_system
    - Type: [Date/Time]
    - Mandatory: [true]
19. Click **Create Output** and fill in the following fields:
    - Label: Expires Local
    - Name: expires_local
    - Type: [Date/Time]
    - Mandatory: [true]
20. Click **Exit Edit Mode**.
21. Under **Action Output**, fill in the following fields:
    - Expiring System: **Script Step > Expiring System**
    - Expiring Local: **Script Step > Expiring Local**
    - Expires System: **Script Step > Expires System**
    - Expires Local: **Script Step > Expires Local**
22. Click **Save**.
23. Click **Publish**.

Break Glass Grant Role Banner

The Break Glass Grant Role Banner is a subflow component within the flow operations. The purpose of the subflow is to setup notification banners for the user to show that they were granted the break glass role, when the break glass role is expiring and when the break glass role has expired.

The subflow uses the ServiceNow OOTB *UX Banner Announcement* feature. ServiceNow banners lack the ability to target individual accounts. To target banners for specific users, users will need unique roles applied to their accounts. This means each user account will need a unique role. This flow checks for the role *u_user_[user id]* to exist. Custom roles are discussed further down.

> **System Property**
>
> Because custom individual roles per user are not standard practice, the subflow checks the system property *break_glass.banner* to see if the value is true or false. If true, then banners are used; else banners are not used. The update set that accompanies this guide sets the value as false.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Subflow**.
3. Under **Subflow properties**, fill in the following fields:
    - Name: Break Glass Grant Role Banner
    - Description:

      ```
      Creates three banners for the break glass role that was granted to the user.

      Note: See system property break_glass.banner to enable or disable this feature..
      ```

    - Run As: [System User]
4. Click **Submit**.
5. Click **More Actions menu** (the ... on top right) and select **Flow Variables**.
6. Click **Add new input** (the +) and fill in the following fields:
    - Label: Banner Label
    - Name: banner_label
    - Type: [String]
7. Click **Add new input** (the +) and fill in the following fields:
    - Label: User Role Name
    - Name: user_role_name
    - Type: [String]
8. Close the **Flow Variables** window.
9. Click **Subflow Inputs & Outputs**.
10. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Request Item
    - Name: request_item
    - Type: (**Reference > Requested Item**)
    - Mandatory: [true]
11. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Break Glass Role
    - Name: break_glass_role
    - Type: (**Reference > Break Glass Role**)
    - Mandatory: [true]
12. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Expiring Local
    - Name: expiring_local
    - Type: [Date/Time]

- Mandatory: [true]

13. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Expires Local
    - Name: expires_local
    - Type: [Date/Time]
    - Mandatory: [true]
14. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Expires
    - Name: expires
    - Type: [Date/Time]
    - Mandatory: [true]
15. Click **Done**.
16. The following instructions are adding operations under the **Actions** section. Due to the difficulty of documenting flow operations, the following sub-ordered steps represent the order of flow operations. At the start of each action there will either be the choice **Add an Action, Flow Logic, or Subflow** or a grouping of **Action**, **Flow Logic**, and **Subflow**. The step will assume general knowledge of which to select and finalizing the operation by clicking **Done**.
    1. **Flow Logic > Set Flow Variables** (click the **+** sign to add a new variable):
        - Name: User Role Name
        - Data: u_user_(**Input > Request Item > Requested for > User ID**)
        - Name: Banner Label
        - Data: break_glass_role.(**Input > Break Glass Role > Name**).(**Input > Request Item > Requested for > User ID**)
    2. **Action > ServiceNow Core > Look Up Records**
        - Table: Banner Announcement [sys_ux_banner_announcement]
        - Conditions:
            - [**Label**] [is] (**Flow Variables > Banner Label**)
        - Don't fail on error: true
    3. **Flow Logic > For Each**
        - Items: (**2 - Lookup Records > Banner Announcement Records**)
    4. (Under *3 - For Each*) **Action > ServiceNow Core > Delete Record**
        - Record: (**3 - For Each > Banner Announcement Record**)
    5. **Action > ServiceNow Core > Look Up Record**
        - Table: System Property [sys_properties]
        - Conditions:
            - [Name] [is] break_glass.banner
    6. **Flow Logic > If**
        - Condition Label: break_glass.banner is false or not found
        - Condition 1:
            - (**5 - Look Up Record > System Property Record > Value**) [is] [false] OR
            - (**5 - Look Up Record > Status**) [is] [Error]
    7. (Under *6 - If*) **Flow Logic > End Subflow**
    8. **Action > ServiceNow Core > Look Up Record**
        - Table: Role [sys_user_role]
        - Conditions:
            - [Name] [is] (**Flow Variables > User Role Name**)

9. **Flow Logic > If**
    - Condition Label: user does not have a private role
    - Condition 1:
        - (**8 - Look Up Record > Status**) [is] [Error]
10. (Under *9 - If*) **Flow Logic > End Subflow**
11. **Action > ServiceNow Core > Create Record**
    - Table: Banner Announcement [sys_ux_banner_announcement]
    - Fields:
        - [Label] (**Flow Variables > Banner Label**)
        - [Heading] Break Glass Role Granted
        - [Summary]

```
var expires = new GlideDateTime(fd_data.subflow_inputs.expires);
var expiresTZ = fd_data.subflow_inputs.request_item.requested_for.time
var tz = Packages.java.util.TimeZone.getTimeZone(expiresTZ);
expires.setTZ(tz);
return "Break glass role '" + fd_data.subflow_inputs.break_glass_role.
```

        - [Color] [Positive]
        - [Show for] [Users with specific roles/groups]
        - [Roles] (**8 - Look Up Record > Role Record > Name**)
        - [Start]

```
var now = new GlideDateTime();
return now.getDisplayValue();
```

        - [End] **Input > Expiring Local**
12. **Action > ServiceNow Core > Create Record**
    - Table: Banner Announcement Mapping [sys_ux_m2m_banner_announcement]
    - Fields:
        - [Announcement Config] [Unified Navigation]
        - [Announcement] (**11 - Create Record > Banner Announcement Record**)
13. **Action > ServiceNow Core > Create Record**
    - Table: Banner Announcement [sys_ux_banner_announcement]
    - Fields:
        - [Label] (**Flow Variables > Banner Label**)
        - [Heading] Break Glass Role Expiring
        - [Summary]

```
var expires = new GlideDateTime(fd_data.subflow_inputs.expires);
var expiresTZ = fd_data.subflow_inputs.request_item.requested_for.time
var tz = Packages.java.util.TimeZone.getTimeZone(expiresTZ);
expires.setTZ(tz);
return "Break glass role '" + fd_data.subflow_inputs.break_glass_role.
```

        - [Color] [Warning]
        - [Show for] [Users with specific roles/groups]
        - [Roles] (**8 - Look Up Record > Role Record > Name**)

        - [Start] **Input > Expiring Local**
        - [End] (**Input > Expires Local**)
    14. **Action > ServiceNow Core > Create Record**
        - Table: Banner Announcement Mapping [sys_ux_m2m_banner_announcement]
        - Fields:
            - [Announcement Config] [Unified Navigation]
            - [Announcement] (**13 - Create Record > Banner Announcement Record**)
    15. **Action > ServiceNow Core > Create Record**
        - Table: Banner Announcement [sys_ux_banner_announcement]
        - Fields:
            - [Label] (**Flow Variables > Banner Label**)
            - [Heading] Break Glass Role Expired
            - [Summary] Break glass role '(**Input > Break Glass Role > Name**)' has expired. Break glass role revocation and user session termination imminent.
            - [Color] [Critical]
            - [Show for] [Users with specific roles/groups]
            - [Roles] (**8 - Look Up Record > Role Record > Name**)
            - [Start] (**Input > Expires Local**)
    16. **Action > ServiceNow Core > Create Record**
        - Table: Banner Announcement Mapping [sys_ux_m2m_banner_announcement]
        - Fields:
            - [Announcement Config] [Unified Navigation]
            - [Announcement] (**15 - Create Record > Banner Announcement Record**)
17. Click **Save**.
18. Click **Publish**.

The above subflow relies on two additional aspects; a system property and that users have a unique user account role. If you do not plan on using banners with your break glass deployment, then set the value of **break_glass.banner** to **false** in the *System Properties [sys_properties]* table.

Instructions:

1. In the **Filter** in the Filter Navigator, enter **sys_properties.list**.
2. Click **New** and fill in the following fields:
    - Name: **break_glass.banner**
    - Description:

```
Enables banner notifications for break glass operations.

Before enabling this, all accounts must have their own private role called u_user_
```

    - Type: [True/False]
    - Value: false
3. Click **Submit**.

> **Banner Activation**

> The steps above default banner activation to false. If you wish to use banner announcements for break glass roles, then set the value to **true**.

> ⚠ **Unique User Roles**
>
> The following instructions will create unique user roles on your ServiceNow instance. If you **do not** intend to use banner announcements for break glass management, then skip to the **Break Glass Grant Role** section below.

The following fix script creates unique roles for each of your existing users on your instance. This is to be used in conjunction with the following business rules as well. The fix script is meant as a one time use script to be executed and then have the business rules maintain the creation and deletion of unique roles for user records going forward.

Instructions:

1. Navigate to **System Definition > Fix Scripts**.
2. Click **New** and fill in the following fields:
   - Name: Break Glass Create Private User Roles
   - Description:

```
The following script creates unique user roles per user account. This is to allow

The fix script is to be used in conjunction with the business rules Grant Unique U
```

   - Script:

```
// note - the code was intentionally commented out to
//        insure you review and confirm its validity
//        in your environment. This is creating
//        unique roles for every account on your
//        instance.

// gather all the user accounts, filtering only those with a user id.
var users = new GlideRecord('sys_user');
users.addEncodedQuery('user_nameISNOTEMPTY');
users.query();
gs.info('Count: ' + users.getRowCount());

/* CODE REVIEW THE FOLLOWING
// go through all the users
while(users.next()) {
    gs.info('User: ' + users.getValue('user_name'));
    var roleName = 'u_user_' + users.getValue('user_name');
    gs.info('\tRole Name: ' + roleName);
    // see if the role already exists
    var roles = new GlideRecord('sys_user_role');
    roles.addQuery('name',roleName);
    roles.query();
```

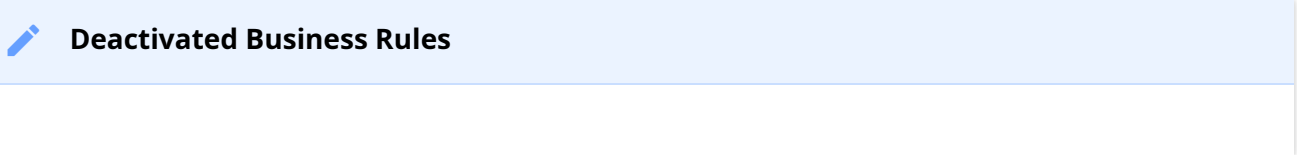
```
    // if the role does not exist, then create the role
    if(!roles.next()) {
        var role = new GlideRecord('sys_user_role');
        role.initialize();
        role.setValue('name',roleName);
        role.insert();
        gs.info('\tRole created.');
    } else {
        gs.info('\tRole exists.');
    }
    // get the role that already existed or was just created
    var uniqueRole = new GlideRecord('sys_user_role');
    uniqueRole.addQuery('name',roleName);
    uniqueRole.query();
    uniqueRole.next();
    if(uniqueRole.getRowCount() > 0) {
        gs.info('\tRole retrieved.');
    } else {
        gs.info('\tWARNING - Role not found.');
    }
    // see if the user has the role already
    var hasRoles = new GlideRecord('sys_user_has_role');
    hasRoles.addQuery('user',users.getValue('sys_id'));
    hasRoles.addQuery('role',uniqueRole.getValue('sys_id'));
    hasRoles.query();
    // if the user does not have the role, then grant the role to the user
    if(!hasRoles.next()) {
        var hasRole = new GlideRecord('sys_user_has_role');
        hasRole.initialize();
        hasRole.setValue('user',users.getValue('sys_id'));
        hasRole.setValue('role',uniqueRole.getValue('sys_id'));
        hasRole.setValue('state','active');
        hasRole.insert();
        gs.info('\tRole granted.');
    } else {
        gs.info('\tRole provided. ' + hasRole.getValue('sys_id'));
    }

}
*/
```

3. In the record header, right-click and select **Save**.

A large section of the code is commented out. This is intentional and meant for you to code review. You are adding unique roles to your instance of ServiceNow. It is important that you make sure that the right accounts will be set up. Once code review is complete, uncomment the code, save the fix script and run the fix script.

The following business rule will create a unique role for new user records.



> The update set that accompanies this guide contains the following deactivated business rules. You will need to activate these business rules, if you wish to use them.

Instructions:

1. Navigate to **System Definition > Business Rules**.
2. Click **New** and fill in the following fields:
    - Name: Create Unique User Role For New Users
    - Table: User [sys_user]
    - Advanced: [true]
3. Under **When to run**, fill in the following fields:
    - When: [after]
    - Insert: [true]
4. Under **Advanced**, fill in the following field:
    - Script:

```
(function executeRule(current, previous /*null when async*/ ) {
    var roleName = 'u_user_' + current.getValue('user_name');
    // see if the role already exists
    var roles = new GlideRecord('sys_user_role');
    roles.addQuery('name', roleName);
    roles.query();
    // if the role does not exist, then create the role
    if (!roles.next()) {
        var role = new GlideRecord('sys_user_role');
        role.initialize();
        role.setValue('name', roleName);
        role.insert();
    }
    // get the role that already existed or was just created
    var uniqueRole = new GlideRecord('sys_user_role');
    uniqueRole.addQuery('name', roleName);
    uniqueRole.query();
    uniqueRole.next();
    // see if the user has the role already
    var hasRoles = new GlideRecord('sys_user_has_role');
    hasRoles.addQuery('user', current.getValue('sys_id'));
    hasRoles.addQuery('role', uniqueRole.getValue('sys_id'));
    hasRoles.query();
    // if the user does not have the role, then grant the role to the user
    if (!hasRoles.next()) {
        var hasRole = new GlideRecord('sys_user_has_role');
        hasRole.initialize();
        hasRole.setValue('user', current.getValue('sys_id'));
        hasRole.setValue('role', uniqueRole.getValue('sys_id'));
        hasRole.setValue('state', 'active');
        hasRole.insert();
    }
})(current, previous);
```

5. Click **Submit**.

The following business rule will delete the unique role when user records are deleted.

Instructions:

1. Navigate to **System Definition > Business Rules**.
2. Click **New** and fill in the following fields:
    - Name: Delete Unique User Role For Deleted User
    - Table: User [sys_user]
    - Advanced: [true]
3. Under **When to run**, fill in the following fields:
    - When: [before]
    - Delete: [true]
4. Under **Advanced**, fill in the following field:
    - Script:

```
(function executeRule(current, previous /*null when async*/ ) {
    var roleName = 'u_user_' + current.getValue('user_name');

    // see if the role exists
    var role = new GlideRecord('sys_user_role');
    role.addQuery('name', roleName);
    role.query();
    // if the role does not exist, then exit
    if (!role.next()) {
        return;
    }

    // remove the role from the user
    var hasRole = new GlideRecord('sys_user_has_role');
    hasRole.addQuery('user', current.getValue('sys_id'));
    hasRole.addQuery('role', role.getValue('sys_id'));
    hasRole.query();
    // if the user does not have the role, then grant the role to the user
    if (hasRole.next()) {
        hasRole.deleteRecord();
    }

    // remove the role
    role.deleteRecord();

})(current, previous);
```

5. Click **Submit**.

Break Glass Grant Role

The *Break Glass Grant Role* is a subflow component within the flow operations. The purpose of the subflow is to grant the user a specific break glass role and setup a banner announcement if enabled to do so.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Subflow**.
3. Under **Subflow properties**, fill in the following fields:
    - Name: Break Glass Grant Role
    - Description: Grants the break glass role to the user.
    - Run As: [System User]
4. Click **Submit**.
5. Click **Subflow Inputs & Outputs**.
6. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Request Item
    - Name: request_item
    - Type: **Reference > Requested Item**
    - Mandatory: [true]
7. Under **Inputs**, click the **+** sign and fill in the following fields:
    - Label: Break Glass Role
    - Name: break_glass_role
    - Type: **Reference > Break Glass Role**
    - Mandatory: [true]
8. Under **Outputs**, click the **+** sign and fill in the following fields:
    - Label: Granted
    - Name: granted
    - Type: [True/False]
9. Under **Outputs**, click the **+** sign and fill in the following fields:
    - Label: Message
    - Name: message
    - Type: [String]
10. Click **Done**.
11. The following instructions are adding operations under the **Actions** section. Due to the difficulty of documenting flow operations, the following sub-ordered steps represent the order of flow operations. At the start of each action there will either be the choice **Add an Action, Flow Logic, or Subflow** or a grouping of **Action**, **Flow Logic**, and **Subflow**. The step will assume general knowledge of which to select and finalizing the operation by clicking **Done**.
    1. **Action > ServiceNow Core > Look Up Records**
        - Table: Break Glass User Role [u_break_glass_user_has_role]
        - Conditions:
            - [Active] [is] [true] AND
            - [Break Glass Role] [is] (**Input > Break Glass Role**) AND
            - [User] [is] (**Input > Request Item > Requested for**)
    2. **Flow Logic > For Each**
        - Items: (**1 - Look Up Records > Break Glass User Role Records**)
    3. (Under 2 - For Each) **Action > ServiceNow Core > Update Record**
        - Record: (**2 - For Each > Break Glass User Role Record**)
        - Table: Break Glass User Role [u_break_glass_user_has_role]
        - Fields:
            - [Active] [false]

4. **Action > ServiceNow Core > Look Up Records**
    - Table: Group Member [sys_user_grmember]
    - Conditions:
        - [Group] [is] (**Input > Break Glass Role > Group**) AND
        - [User] [is] (**Input > Request Item > Requested for**)
5. **Flow Logic > If**
    - Condition Label: the user is not a member of the group
    - Condition 1:
        - (**4 - Look Up Records > Count**) [is] 0
6. (Under 5 - If) **Action > ServiceNow Core > Create Record**
    - Table: Group Member [sys_user_grmember]
    - Fields:
        - [Group] (**Input > Break Glass Role > Group**)
        - [User] (**Input > Request Item > Requested for**)
7. **Action > Global > Break Glass Get Expirations**
    - Break Glass Role: **Input > Break Glass Role**
8. **Action > ServiceNow Core > Create Record**
    - Table: Break Glass User Role [u_break_glass_user_has_role]
    - Fields:
        - [User] (**Input > Request Item > Requested for**)
        - [Break Glass Role] (**Input > Break Glass Role**)
        - [Active] [true]
        - [Expiration] (**7 - Break Glass Get Expirations > Expires Local**)
        - [Request Item] (**Input > Request Item**)
9. **Subflow > Break Glass Grant Role Banner**
    - Request Item: (**Input > Request Item**)
    - Break Glass Role: (**Input > Break Glass Role**)
    - Expiring Local: (**7 - Break Glass Get Expirations > Expiring Local**)
    - Expires Local: (**7 - Break Glass Get Expirations > Expires Local**)
    - Expires: (**7 - Break Glass Get Expirations > Expires System**)
10. **Flow Logic > Assign Subflow Outputs**
    - [Granted] [true]
    - [Message] Granted the break glass role ,(**Input > Break Glass Role > Name**)' to (**Input > Request Item > Requested for > Name**).
11. Toggle **ERROR HANDLER** to **true**.
12. (Under ERROR HANDLER) **Flow Logic > Assign Subflow Outputs**
    - [Granted] [false]
    - [Message] An unexpected error occurred granting the break glass role '(**Input > Break Glass Role > Name**)' to (**Input > Request Item > Requested for > Name**).
12. Click **Save**.
13. Click **Publish**.

Break Glass Close User Sessions

The *Break Glass Close User Sessions* is an action component within the flow operations. The purpose of

the action is to terminate all of the user's active sessions. This will force the user to re-login into ServiceNow and the user's account rolled privileges reset. This is also used later on, when automation revokes the break glass role from the user as well.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Action**.
3. Fill in the following fields: -Action name: Break Glass Close User Sessions -Description: Terminate the user's active ServiceNow sessions forcing the user to have to re-login into ServiceNow.
4. Click **Submit**.
5. Under **Action Outline**, click **Inputs**.
6. In the **Action Inputs** section, click **Create Input** and fill in the following fields:
    - Label: User
    - Name: user
    - Type: **Reference > User [sys_user]**
    - Mandatory: [true]
7. Under **Action Outline**, between **Inputs** and **Error Evaluation**, click the **+** sign.
8. In the **Choose a step to add to your action**, find and select **Script**.
9. Under **Script Step**, under **Input Variables**, click **Create Variable**, and fill in the following fields:
    - Name: user_id
    - Value: (**Inputs > User > User ID**)
10. Under **Script** add the following:

```
(function execute(inputs, outputs) {
  GlideSessions.lockOutSessionsInAllNodes(inputs.user_id);
})(inputs, outputs);
```

11. Click **Save**.
12. Click **Publish**.

Break Glass Roles - Catalog Item

The *Break Glass Roles - Catalog Item* is the flow that processes service requests from the *Break Glass Roles* catalog item.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Flow**.
3. In the **Flow properties** window, fill in the following fields:
    - Flow name: Break Glass Roles - Catalog Item
    - Description: Processes Break Glass Roles catalog item service requests.
    - Application: [Global]
    - Run As: [System User]

4. Click **Submit**.
5. Click **More Actions menu** (the ... on top right) and select **Flow Variables**.
6. Click **Add new input** (the +) and fill in the following fields:
    - Label: Granted Role
    - Name: granted_role
    - Type: [True/False]
7. Click **Add new input** (the +) and fill in the following fields:
    - Label: Failure Detected
    - Name: failure_detected
    - Type: [True/False]
8. Close the **Flow Variables** window.
9. Under **Trigger**, click **Add a trigger** and under **Application** select **Service Catalog**.
10. Click **Done**.
11. The following instructions are adding operations under the **Actions** section. Due to the difficulty of documenting flow operations, the following sub-ordered steps represent the order of flow operations. At the start of each action there will either be the choice **Add an Action, Flow Logic, or Subflow** or a grouping of **Action**, **Flow Logic**, and **Subflow**. The step will assume general knowledge of which to select and finalizing the operation by clicking **Done**.
    1. **Flow Logic > Set Flow Variables**
        - Name: Granted Role
        - Data: [false]
        - Name: Failure Detected
        - Data: [false]
    2. **Action > ServiceNow Core > Get Catalog Variables**, fill in the following fields:
        - Submitted Request: (**Trigger - Service Catalog > Requested Item Record**)
        - Template Catalog Items and Variables Sets [Catalog Items and Variable Sets]: [Break Glass Roles]
        - Catalog Variables: {Under **Available** list click **roles** and then **>** to move to **Selected**.}
    3. **Action > ServiceNow Core > Look Up Records**
        - Table: Break Glass Role [u_break_glass_role]
        - Conditions:
            - [Sys ID] [is one of] (**2 - Get Catalog Variables > roles**)
    4. **Flow Logic > For Each**
        - Items: (**3 - Look Up Records > Break Glass Role Records**)
    5. (Under 4 - For Each) **Subflow > Break Glass Grant Role**
        - Wait For Completion: [true]
        - Request Item: (**Trigger - Service Catalog > Requested Item Record**)
        - Break Glass Role: (**4 - For Each > Break Glass Role Record**)
    6. (Under 4 - For Each) **Action > ServiceNow Core > Update Record**
        - Record: (**Trigger - Service Catalog > Requested Item Record**)
        - Table: Requested Item [sc_rec_item]
        - Fields:
            - [Additional comments] (**5 - Break Glass Grant Role > Message**)
    7. (Under 4 - For Each) **Flow Logic > If**
        - Condition Label: a break glass role was granted

        - Condition 1: (**5 - Break Glass Grant Role > Granted**) [is] [True]
    8. (Under 7 - If) **Flow Logic > Set Flow Variables**
        - Name: Granted Role
        - Data: [true]
    9. (Under 7 - If) **Flow Logic > Else**
    10. (Under 9 - Else) **Flow Logic > Set Flow Variables**
        - Name: Failure Detected
        - Data: [true]
    11. **Flow Logic > If**
        - Condition Label: a break glass role was granted
        - Condition 1: (**Flow Variables > Granted Role**) [is] [True]
    12. (Under 11 - If) **Action > Global > Break Glass Close User Sessions**
        - User [User] (**Trigger - Service Catalog > Requested Item Record > Requested for**)
    13. **Flow Logic > If**
        - Condition Label: a failure was detected
        - Condition 1: (**Flow Variables > Failure Detected**) [is] [True]
    14. (Under 13 - If) **Action > ServiceNow Core > Update Record**
        - Record: (**Trigger - Service Catalog > Requested Item Record**)
        - Table: Requested Item [sc_req_item]
        - Fields:
            - [Short description] Failure Detected - (**Trigger - Service Catalog > Requested Item Record > Short description**)
            - [Additional comments] Automation detected a failure in granting the break glass roles to the account. Investigate the issue and resolve.
    15. **Flow Logic > Else**
    16. **Action > ServiceNow Core > Update Record**
        - Record: (**Trigger - Service Catalog > Requested Item Record**)
        - Table: Requested Item [sc_req_item]
        - Fields:
            - [Close notes] Completed granting the break glass roles to the user. No failures were detected.
            - [State] [Closed Complete]
12. Click **Save**.
13. Click **Activate**.

With the *Break Glass Roles - Catalog Item* flow complete, the catalog item *Break Glass Roles* needs to be updated.

Instructions:

1. Navigate to **Service Catalog > Catalog Definitions > Maintain Items**.
2. Search for and open the **Break Glass Roles** catalog item.
3. Under the **Process Engine** tab, fill in the following field:
    - Flow: [Break Glass Roles - Catalog Item]
4. Click **Update**.

Break Glass Role Revocation

The break glass revocation process that removes the role(s) from the user's account is handled by a repeating flow that runs every minute. The frequency of execution is arbituary, tune as warranted.

Instructions:

1. Navigate to **Process Automation > Flow Designer**. A new tab will open for Flow Designer.
2. In **Flow Designer** click **New** and select **Flow**.
3. Under **Flow properties**, fill in the following fields:
    - Name: Break Glass Roles - Repeating Job
    - Description: Revokes break glass roles.
    - Run As: [System User]
4. Click **Submit**.
5. Click **More Actions menu** (the ... on top right) and select **Flow Variables**.
6. Click **Add new input** (the +) and fill in the following fields:
    - Label: Banner Label
    - Name: banner_label
    - Type: [String]
7. Close the **Flow Variables** window.
8. Under **Trigger**, click **Add a trigger**, select **Repeat** and fill in the following field:
    - Days: 0
    - h: 0
    - m: 1
    - s: 0
9. Click **Done**.
10. The following instructions are adding operations under the **Actions** section. Due to the difficulty of documenting flow operations, the following sub-ordered steps represent the order of flow operations. At the start of each action there will either be the choice **Add an Action, Flow Logic, or Subflow** or a grouping of **Action**, **Flow Logic**, and **Subflow**. The step will assume general knowledge of which to select and finalizing the operation by clicking **Done**.
    1. **Action > ServiceNow Core > Look Up Records**
        - Table: Break Glass User Role [u_break_glass_user_has_role]
        - _Conditions:
            - [Active] [is] [true] AND
            - [Expiration] [at or before] (**Trigger - Repeat > Run Start Time UTC**)
    2. **Flow Logic > For Each**
        - Items: (**1 - Look Up Records > Break Glass User Role Records**)
    3. (Under 2 - For Each) **Flow Logic > Set Flow Variables**
        - Name: Banner Label
        - Data: break_glass_role.(**2 - For Each > Break Glass User Role Record > Break Glass Role > Name**).(**2 - For Each > Break Glass User Role Record > User > User ID**)
    4. (Under 2 - For Each) **Action > ServiceNow Core > Look Up Records**
        - Table: Group Member [sys_user_grmember]
        - Conditions:

        - [Group] [is] (**2 - For Each > Break Glass User Role Record > Break Glass Role > Group**) AND
        - [User] [is] (**2 - For Each > Break Glass User Role Record > User**)
    5. (Under 2 - For Each) **Flow Logic > For Each**
        - Items: (**4 - Look Up Records > Group Member Records**)
    6. (Under 5 - For Each) **Action > ServiceNow Core > Delete Record**
        - Record: (**5 - For Each > User Role Record**)
    7. (Under 2 - For Each) **Action > ServiceNow Core > Look Up Records**
        - Table: Banner Announcement [sys_ux_banner_announcement]
        - Conditions:
            - [Label] [is] (**Flow Variables > Banner Label**)
    8. (Under 2 - For Each) **Flow Logic > For Each**
        - Items: (**7 - Look Up Records > Banner Announcement Records**)
    9. (Under 8 - For Each) **Action > ServiceNow Core > Delete Record**
        - Record: (**8 - For Each > Banner Announcement Record**)
    10. (Under 2 - For Each) **Action > ServiceNow Core > Update Record**
        - Record: (**2 - For Each > Break Glass User Role Record**)
        - Table: Break Glass User Role [u_break_glass_user_has_role]
        - Fields:
            - [Active] false
    11. (Under 2 - For Each) **Action > ServiceNow Core > Update Record**
        - Record: (**2 - For Each > Break Glass User Role Record > Request item**)
        - Table: Requested Item [sc_req_item]
        - Fields:
            - [Additional Comments] Revoked break glass role '(**2 - For Each > Break Glass User Role Record > Break Glass Role > Name**)'.
    12. (Under 2 - For Each) **Action > Global > Break Glass Close User Sessions**
        - User: (**2 - For Each > Break Glass User Role Record > User**)
11. Click **Save**.
12. Click **Activate**.

Next Steps

With BGRM setup, the next steps will be to configure the break glass roles.

Instructions:

1. If you have not already, create the ServiceNow roles that will be used. Standard OOTB roles can be used.
2. If you have not already, create the ServiceNow groups that will be used and assign the appropriate roles to the groups.
3. Navigate to **User Administration > Break Glass Roles** and create new break glass role records.

> **Deactivated Groups**

> As a reminder, you can deactivate groups preventing users from selecting break glass roles where groups are deactivated.

Outro

You have now successfully setup a break glass role management framework. Congratulations.