steveiliop56 / **tinyauth** Public

Code | Issues 14 | Pull requests 7 | Discussions | Actions | Projects

Commit `f26c217`

steveiliop56 authored 2 weeks ago · ✓ 6 / 6 · Verified

```
refactor: oauth flow (#726)

* wip

* feat: add oauth session impl in auth service

* feat: move oauth logic into auth service and handle multiple sessions

* tests: fix tests

* fix: review comments

* fix: prevent ddos attacks in oauth rate limit
```

`main` (#726) · v5.0.6 ••• nightly

1 parent `d71a8e0` commit `f26c217`

15 files changed +519 -557 lines changed

Top

Filter files…

- internal
  - bootstrap
    - app_bootstrap.go
    - router_bootstrap.go
    - service_bootstrap.go
  - config
    - config.go
  - controller
    - oauth_controller.go

- proxy_controller_test.go
- user_controller_test.go
- service
  - auth_service.go
  - generic_oauth_service.go
  - github_oauth_service.go
  - google_oauth_service.go
  - oauth_broker_service.go
  - oauth_extractors.go
  - oauth_presets.go
  - oauth_service.go

15 files changed +519 -557 lines changed

Search within code

internal/bootstrap/app_bootstrap.go

```
@@ -22,16 +22,17 @@ import (
22 22  type BootstrapApp struct {
23 23      config  config.Config
24 24      context struct {
25    -         appUrl             string
26    -         uuid               string
27    -         cookieDomain       string
28    -         sessionCookieName  string
29    -         csrfCookieName     string
30    -         redirectCookieName string
31    -         users              []config.User
32    -         oauthProviders     map[string]config.OAuthServiceConfig
33    -         configuredProviders []controller.Provider
34    -         oidcClients        []config.OIDCClientConfig
   25 +         appUrl                string
   26 +         uuid                  string
   27 +         cookieDomain          string
   28 +         sessionCookieName     string
   29 +         csrfCookieName        string
   30 +         redirectCookieName    string
   31 +         oauthSessionCookieName string
```

```
      32 +          users                  []config.User
      33 +          oauthProviders         map[string]config.OAuthServiceConfig
      34 +          configuredProviders    []controller.Provider
      35 +          oidcClients            []config.OIDCClientConfig
35    36           }
36    37           services Services
37    38       }
         ↕    @@ -113,6 +114,7 @@ func (app *BootstrapApp) Setup() error {
113   114          app.context.sessionCookieName = fmt.Sprintf("%s-%s",
                   config.SessionCookieName, cookieId)
114   115          app.context.csrfCookieName = fmt.Sprintf("%s-%s", config.CSRFCookieName,
                   cookieId)
115   116          app.context.redirectCookieName = fmt.Sprintf("%s-%s",
                   config.RedirectCookieName, cookieId)
      117 +      app.context.oauthSessionCookieName = fmt.Sprintf("%s-%s",
                   config.OAuthSessionCookieName, cookieId)
116   118
117   119          // Dumps
118   120          tlog.App.Trace().Interface("config", app.config).Msg("Config dump")
         ↕    @@ -190,12 +192,12 @@ func (app *BootstrapApp) Setup() error {
190   192
191   193          // Start db cleanup routine
192   194          tlog.App.Debug().Msg("Starting database cleanup routine")
193       -      go app.dbCleanup(queries)
      195 +      go app.dbCleanupRoutine(queries)
194   196
195   197          // If analytics are not disabled, start heartbeat
196   198          if app.config.Analytics.Enabled {
197   199              tlog.App.Debug().Msg("Starting heartbeat routine")
198       -          go app.heartbeat()
      200 +          go app.heartbeatRoutine()
199   201          }
200   202
201   203          // If we have an socket path, bind to it
         ↕    @@ -226,7 +228,7 @@ func (app *BootstrapApp) Setup() error {
226   228          return nil
227   229      }
```

```
228  230
229       - func (app *BootstrapApp) heartbeat() {
     231  + func (app *BootstrapApp) heartbeatRoutine() {
230  232       ticker := time.NewTicker(time.Duration(12) * time.Hour)
231  233       defer ticker.Stop()
232  234
          @@ -280,7 +282,7 @@ func (app *BootstrapApp) heartbeat() {
280  282       }
281  283   }
282  284
283       - func (app *BootstrapApp) dbCleanup(queries *repository.Queries) {
     285  + func (app *BootstrapApp) dbCleanupRoutine(queries *repository.Queries) {
284  286       ticker := time.NewTicker(time.Duration(30) * time.Minute)
285  287       defer ticker.Stop()
286  288       ctx := context.Background()
```

internal/bootstrap/router_bootstrap.go

```
          @@ -77,12 +77,13 @@ func (app *BootstrapApp) setupRouter() (*gin.Engine,
          error) {
77  77        contextController.SetupRoutes()
78  78
79  79        oauthController :=
          controller.NewOAuthController(controller.OAuthControllerConfig{
80        -         AppURL:             app.config.AppURL,
81        -         SecureCookie:       app.config.Auth.SecureCookie,
82        -         CSRFCookieName:     app.context.csrfCookieName,
83        -         RedirectCookieName: app.context.redirectCookieName,
84        -         CookieDomain:       app.context.cookieDomain,
85        -     }, apiRouter, app.services.authService, app.services.oauthBrokerService)
      80  +         AppURL:                 app.config.AppURL,
      81  +         SecureCookie:           app.config.Auth.SecureCookie,
      82  +         CSRFCookieName:         app.context.csrfCookieName,
      83  +         RedirectCookieName:     app.context.redirectCookieName,
      84  +         CookieDomain:           app.context.cookieDomain,
      85  +         OAuthSessionCookieName: app.context.oauthSessionCookieName,
      86  +     }, apiRouter, app.services.authService)
86  87
87  88        oauthController.SetupRoutes()
```

```
88  89
```

internal/bootstrap/service_bootstrap.go

```
@@ -58,6 +58,16 @@ func (app *BootstrapApp) initServices(queries
*repository.Queries) (Services, er
58  58
59  59          services.accessControlService = accessControlsService
60  60
+   61      oauthBrokerService :=
+       service.NewOAuthBrokerService(app.context.oauthProviders)
+   62
+   63      err = oauthBrokerService.Init()
+   64
+   65      if err != nil {
+   66          return Services{}, err
+   67      }
+   68
+   69      services.oauthBrokerService = oauthBrokerService
+   70
61  71          authService := service.NewAuthService(service.AuthServiceConfig{
62  72              Users:              app.context.users,
63  73              OauthWhitelist:     app.config.OAuth.Whitelist,
@@ -70,7 +80,7 @@ func (app *BootstrapApp) initServices(queries
*repository.Queries) (Services, er
70  80              SessionCookieName:  app.context.sessionCookieName,
71  81              IP:                 app.config.Auth.IP,
72  82              LDAPGroupsCacheTTL: app.config.Ldap.GroupCacheTTL,
-   73      }, dockerService, services.ldapService, queries)
+   83      }, dockerService, services.ldapService, queries,
+       services.oauthBrokerService)
74  84
75  85          err = authService.Init()
76  86
@@ -80,16 +90,6 @@ func (app *BootstrapApp) initServices(queries
*repository.Queries) (Services, er
80  90
81  91          services.authService = authService
82  92
```

```
83        -     oauthBrokerService :=
          service.NewOAuthBrokerService(app.context.oauthProviders)
84        -
85        -     err = oauthBrokerService.Init()
86        -
87        -     if err != nil {
88        -         return Services{}, err
89        -     }
90        -
91        -     services.oauthBrokerService = oauthBrokerService
92        -
93  93          oidcService := service.NewOIDCService(service.OIDCServiceConfig{
94  94              Clients:        app.config.OIDC.Clients,
95  95              PrivateKeyPath: app.config.OIDC.PrivateKeyPath,
```

internal/config/config.go

```
    @@ -73,6 +73,7 @@ var BuildTimestamp = "0000-00-00T00:00:00Z"
73  73  var SessionCookieName = "tinyauth-session"
74  74  var CSRFCookieName = "tinyauth-csrf"
75  75  var RedirectCookieName = "tinyauth-redirect"
    76  + var OAuthSessionCookieName = "tinyauth-oauth"
76  77
77  78  // Main app config
78  79
```

internal/controller/oauth_controller.go

```
    @@ -21,26 +21,25 @@ type OAuthRequest struct {
21  21  }
22  22
23  23  type OAuthControllerConfig struct {
24      -     CSRFCookieName     string
25      -     RedirectCookieName string
26      -     SecureCookie       bool
27      -     AppURL             string
28      -     CookieDomain       string
    24  +     CSRFCookieName         string
    25  +     OAuthSessionCookieName string
```

```
      26  +     RedirectCookieName     string
      27  +     SecureCookie           bool
      28  +     AppURL                 string
      29  +     CookieDomain           string
29    30      }
30    31
31    32      type OAuthController struct {
32    33          config OAuthControllerConfig
33    34          router *gin.RouterGroup
34    35          auth   *service.AuthService
35          -     broker *service.OAuthBrokerService
36    36      }
37    37
38          - func NewOAuthController(config OAuthControllerConfig, router *gin.RouterGroup,
              auth *service.AuthService, broker *service.OAuthBrokerService) *OAuthController
              {
      38  + func NewOAuthController(config OAuthControllerConfig, router *gin.RouterGroup,
              auth *service.AuthService) *OAuthController {
39    39          return &OAuthController{
40    40              config: config,
41    41              router: router,
42    42              auth:   auth,
43          -         broker: broker,
44    43          }
45    44      }
46    45
              @@ -63,21 +62,30 @@ func (controller *OAuthController) oauthURLHandler(c
              *gin.Context) {
63    62              return
64    63          }
65    64
66          -     service, exists := controller.broker.GetService(req.Provider)
      65  +     sessionId, session, err := controller.auth.NewOAuthSession(req.Provider)
67    66
68          -     if !exists {
69          -         tlog.App.Warn().Msgf("OAuth provider not found: %s", req.Provider)
70          -         c.JSON(404, gin.H{
71          -             "status":  404,
72          -             "message": "Not Found",
      67  +     if err != nil {
```

```
68  +         tlog.App.Error().Err(err).Msg("Failed to create OAuth session")
69  +         c.JSON(500, gin.H{
70  +             "status":  500,
71  +             "message": "Internal Server Error",
72  +         })
73  +         return
74  +     }
75  +
76  +     authUrl, err := controller.auth.GetOAuthURL(sessionId)
77  +
78  +     if err != nil {
79  +         tlog.App.Error().Err(err).Msg("Failed to get OAuth URL")
80  +         c.JSON(500, gin.H{
81  +             "status":  500,
82  +             "message": "Internal Server Error",
83      83             })
84      84             return
85      85         }
86      86
77      -     service.GenerateVerifier()
78      -     state := service.GenerateState()
79      -     authURL := service.GetAuthURL(state)
80      -     c.SetCookie(controller.config.CSRFCookieName, state,
              int(time.Hour.Seconds()), "/", fmt.Sprintf(".%s",
              controller.config.CookieDomain), controller.config.SecureCookie, true)
87  +     c.SetCookie(controller.config.OAuthSessionCookieName, sessionId,
          int(time.Hour.Seconds()), "/", fmt.Sprintf(".%s",
          controller.config.CookieDomain), controller.config.SecureCookie, true)
88  +     c.SetCookie(controller.config.CSRFCookieName, session.State,
          int(time.Hour.Seconds()), "/", fmt.Sprintf(".%s",
          controller.config.CookieDomain), controller.config.SecureCookie, true)
89      89
90      90         redirectURI := c.Query("redirect_uri")
91      91         isRedirectSafe := utils.IsRedirectSafe(redirectURI,
              controller.config.CookieDomain)
          @@ -95,7 +103,7 @@ func (controller *OAuthController) oauthURLHandler(c
          *gin.Context) {
103     103         c.JSON(200, gin.H{
104     104             "status":  200,
105     105             "message": "OK",
```

```
98        -         "url":     authURL,
      106 +         "url":     authUrl,
99    107         })
100   108     }
101   109
          @@ -112,6 +120,17 @@ func (controller *OAuthController)
          oauthCallbackHandler(c *gin.Context) {
112   120             return
113   121         }
114   122
      123 +     sessionIdCookie, err := c.Cookie(controller.config.OAuthSessionCookieName)
      124 +
      125 +     if err != nil {
      126 +         tlog.App.Warn().Err(err).Msg("OAuth session cookie missing")
      127 +         c.Redirect(http.StatusTemporaryRedirect, fmt.Sprintf("%s/error",
          controller.config.AppURL))
      128 +         return
      129 +     }
      130 +
      131 +     c.SetCookie(controller.config.OAuthSessionCookieName, "", -1, "/",
          fmt.Sprintf(".%s", controller.config.CookieDomain),
          controller.config.SecureCookie, true)
      132 +     defer controller.auth.EndOAuthSession(sessionIdCookie)
      133 +
115   134         state := c.Query("state")
116   135         csrfCookie, err := c.Cookie(controller.config.CSRFCookieName)
117   136
          @@ -125,28 +144,15 @@ func (controller *OAuthController)
          oauthCallbackHandler(c *gin.Context) {
125   144         c.SetCookie(controller.config.CSRFCookieName, "", -1, "/",
          fmt.Sprintf(".%s", controller.config.CookieDomain),
          controller.config.SecureCookie, true)
126   145
127   146         code := c.Query("code")
128       -     service, exists := controller.broker.GetService(req.Provider)
      147 +     _, err = controller.auth.GetOAuthToken(sessionIdCookie, code)
129   148
130       -     if !exists {
131       -         tlog.App.Warn().Msgf("OAuth provider not found: %s", req.Provider)
```

```
132      -          c.Redirect(http.StatusTemporaryRedirect, fmt.Sprintf("%s/error",
         controller.config.AppURL))
133      -          return
134      -      }
135      -
136      -      err = service.VerifyCode(code)
137  149          if err != nil {
138      -          tlog.App.Error().Err(err).Msg("Failed to verify OAuth code")
     150  +          tlog.App.Error().Err(err).Msg("Failed to exchange code for token")
139  151              c.Redirect(http.StatusTemporaryRedirect, fmt.Sprintf("%s/error",
         controller.config.AppURL))
140  152              return
141  153          }
142  154
143      -      user, err := controller.broker.GetUser(req.Provider)
144      -
145      -      if err != nil {
146      -          tlog.App.Error().Err(err).Msg("Failed to get user from OAuth provider")
147      -          c.Redirect(http.StatusTemporaryRedirect, fmt.Sprintf("%s/error",
         controller.config.AppURL))
148      -          return
149      -      }
     155  +      user, err := controller.auth.GetOAuthUserinfo(sessionIdCookie)
150  156
151  157          if user.Email == "" {
152  158              tlog.App.Error().Msg("OAuth provider did not return an email")
         @@ -192,13 +198,21 @@ func (controller *OAuthController)
         oauthCallbackHandler(c *gin.Context) {
192  198              username = strings.Replace(user.Email, "@", "_", 1)
193  199          }
194  200
     201  +      service, err := controller.auth.GetOAuthService(sessionIdCookie)
     202  +
     203  +      if err != nil {
     204  +          tlog.App.Error().Err(err).Msg("Failed to get OAuth service for
         session")
     205  +          c.Redirect(http.StatusTemporaryRedirect, fmt.Sprintf("%s/error",
         controller.config.AppURL))
     206  +          return
     207  +      }
```

```
208 +
195 209         sessionCookie := repository.Session{
196 210             Username:    username,
197 211             Name:        name,
198 212             Email:       user.Email,
199 213             Provider:    req.Provider,
200 214             OAuthGroups: utils.CoalesceToString(user.Groups),
201     -           OAuthName:   service.GetName(),
    215 +           OAuthName:   service.Name(),
202 216             OAuthSub:    user.Sub,
203 217         }
204 218
```

internal/controller/proxy_controller_test.go

```
@@ -85,7 +85,7 @@ func setupProxyController(t *testing.T, middlewares
[]gin.HandlerFunc) (*gin.Eng
85 85             LoginTimeout:       300,
86 86             LoginMaxRetries:    3,
87 87             SessionCookieName:  "tinyauth-session",
88     -       }, dockerService, nil, queries)
    88 +       }, dockerService, nil, queries, &service.OAuthBrokerService{})
89 89
90 90         // Controller
91 91         ctrl := controller.NewProxyController(controller.ProxyControllerConfig{
```

internal/controller/user_controller_test.go

```
@@ -71,7 +71,7 @@ func setupUserController(t *testing.T, middlewares *
[]gin.HandlerFunc) (*gin.Eng
71 71             LoginTimeout:       300,
72 72             LoginMaxRetries:    3,
73 73             SessionCookieName:  "tinyauth-session",
74     -       }, nil, nil, queries)
    74 +       }, nil, nil, queries, &service.OAuthBrokerService{})
75 75
76 76         // Controller
77 77         ctrl := controller.NewUserController(controller.UserControllerConfig{
```

Comments 0

Please sign in to comment.