~sircmpwn/sourcehut

1ea24685a76f6456245d0bc7fe6accf6231c4044 — Drew DeVault 807829e 0.56.11 5 years ago

```
Move account registration into GQL

This moves the account registration process into GraphQL and updates
meta.sr.ht to be a thin shim on top of it. This also adds a database
constraint to prevent duplicate emails - thankfully there was only one.

This change also fixes a couple of lingering bugs: attempting to sign up
with a PGP key already in the system now shows an error, and a few
places where key_id was still used in the UI were replaced with
fingerprint_hex.
```

patch ▸

browse ▸

.tar.gz ▸

.zip ▸

```
17 files changed, 1748 insertions(+), 217 deletions(-)

M api/go.mod
M api/go.sum
M api/graph/api/generated.go
A api/graph/blacklist.go
M api/graph/resolver.go
M api/graph/schema.graphqls
M api/graph/schema.resolvers.go
M api/server.go
A metasrht/alembic/versions/8928d88c66d7_add_more_constraints_for_users.py
M metasrht/auth/builtin.py
D metasrht/auth_validation.py
M metasrht/blueprints/auth.py
M metasrht/blueprints/profile.py
M metasrht/templates/privacy.html
M metasrht/templates/register-step2.html
M metasrht/templates/register.html
M metasrht/templates/user.html
```

M api/go.mod => api/go.mod +2 -1

```
@@ 3,7 3,7 @@ module git.sr.ht/~sircmpwn/meta.sr.ht/api
 go 1.14

 require (
-	git.sr.ht/~sircmpwn/core-go v0.0.0-20210909084213-468752564125 // indirect
+	git.sr.ht/~sircmpwn/core-go v0.0.0-20210924112641-378fedbc638f // indirect
	git.sr.ht/~sircmpwn/dowork v0.0.0-20210820133136-d3970e97def3 // indirect
	git.sr.ht/~sircmpwn/go-bare v0.0.0-20210227202403-5dae5c48f917 // indirect
	github.com/99designs/gqlgen v0.13.0


@@ 23,6 23,7 @@ require (
	github.com/kr/text v0.2.0 // indirect
	github.com/lib/pq v1.8.0
	github.com/mitchellh/mapstructure v1.3.2 // indirect
+	github.com/nbutton23/zxcvbn-go v0.0.0-20210217022336-fa2cb2858354 // indirect
	github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e // indirect
	github.com/prometheus/common v0.30.0 // indirect
	github.com/prometheus/procfs v0.7.3 // indirect
```

M api/go.sum => api/go.sum +7 -0

```
@@ 62,6 62,10 @@ git.sr.ht/~sircmpwn/core-go v0.0.0-20210901140702-d74ae98eb367 h1:Fofz0gEsnt4l4c
 git.sr.ht/~sircmpwn/core-go v0.0.0-20210901140702-d74ae98eb367/go.mod
h1:dOzMmMQFPH0ztBhLFNo/hFLHqck1tbhgL3aNi1XnOsI=
 git.sr.ht/~sircmpwn/core-go v0.0.0-20210909084213-468752564125 h1:UbY+U6d65kx09HhIDkakHd55uGb9SSVta76uyHjqKW8=
 git.sr.ht/~sircmpwn/core-go v0.0.0-20210909084213-468752564125/go.mod
h1:dOzMmMQFPH0ztBhLFNo/hFLHqck1tbhgL3aNi1XnOsI=
+git.sr.ht/~sircmpwn/core-go v0.0.0-20210924111216-3b553750746e h1:+5NgeBztmqDkcRXVJNTKbdmXTEdNjIGTxsk+omg2KIE=
+git.sr.ht/~sircmpwn/core-go v0.0.0-20210924111216-3b553750746e/go.mod
h1:dOzMmMQFPH0ztBhLFNo/hFLHqck1tbhgL3aNi1XnOsI=
+git.sr.ht/~sircmpwn/core-go v0.0.0-20210924112641-378fedbc638f h1:JlJnutvLZHgNjwf5Lszrs7Slcjr1AVDUIV0Jm/PpRWg=
+git.sr.ht/~sircmpwn/core-go v0.0.0-20210924112641-378fedbc638f/go.mod
h1:dOzMmMQFPH0ztBhLFNo/hFLHqck1tbhgL3aNi1XnOsI=
 git.sr.ht/~sircmpwn/dowork v0.0.0-20201013160733-35ca012e4dc8/go.mod h1:8neHEO3503w/rNtttnR0JFpQgM/GFhaafVwvkPsFIDw=
 git.sr.ht/~sircmpwn/dowork v0.0.0-20201121170652-c2a771442daf h1:wRE9o+wlpTSuq/ucFJsfbglAobfGPIgFdaTtZXrk8P0=
 git.sr.ht/~sircmpwn/dowork v0.0.0-20201121170652-c2a771442daf/go.mod h1:8neHEO3503w/rNtttnR0JFpQgM/GFhaafVwvkPsFIDw=


@@ 477,6 481,8 @@ github.com/nats-io/nkeys v0.1.3 h1:6JrEfig+HzTH85yxzhSVbjHRJv9cn0p6n3IngIcM5/k=
 github.com/nats-io/nkeys v0.1.3/go.mod h1:xpnFELMwJABBLVhffcfd1MZx6VsNRFpEugbxziKVo7w=
 github.com/nats-io/nuid v1.0.1 h1:5iA8DT8V7q8WK2EScv2padNa/rTESc1KdnPw4TC2paw=
 github.com/nats-io/nuid v1.0.1/go.mod h1:19wcPz3Ph3q0Jbyiqsd0kePYG7A95tJPxeL+1OSON2c=
+github.com/nbutton23/zxcvbn-go v0.0.0-20210217022336-fa2cb2858354 h1:4kuARK6Y6FxaNu/BnU2OAaLF86eTVhP2hjTB6iMvItA=
+github.com/nbutton23/zxcvbn-go v0.0.0-20210217022336-fa2cb2858354/go.mod
h1:KSVJerMDfblTH7p5MZaTt+8zaT2iEk3AkVb9PQdZuE8=
 github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e h1:fD57ERR4JtEqsWbfPhv4DMiApHyliiK5xCTNVSPiaAs=
 github.com/niemeyer/pretty v0.0.0-20200227124842-a10e7caefd8e/go.mod h1:zD1mROLANZcx1PVRCS0qkT7pwLkGfwJo4zjcN/Tysno=
 github.com/nxadm/tail v1.4.4 h1:DQuhQpB1tVlglWS2hLQ5OV6B5r8aGxSrPc5Qo6uTN78=


@@ 618,6 624,7 @@ github.com/streadway/handy v0.0.0-20190108123426-d5acb3125c2a/go.mod h1:qNTQ5P5J
 github.com/stretchr/objx v0.1.0/go.mod h1:HFkY916IF+rwdDfMAkV7OtwuqBVzrE8GR6GFx+wExME=
 github.com/stretchr/objx v0.1.1 h1:2vfRuCMp5sSVIDSqO8oNnWJq7mPa6KVP3iPIwFBuy8A=
 github.com/stretchr/objx v0.1.1/go.mod h1:HFkY916IF+rwdDfMAkV7OtwuqBVzrE8GR6GFx+wExME=
+github.com/stretchr/testify v1.1.4/go.mod h1:a8OnRcib4nhh0OaRAV+Yts87kKdq0PP7pXfy6kDkUVs=
 github.com/stretchr/testify v1.2.1/go.mod h1:a8OnRcib4nhh0OaRAV+Yts87kKdq0PP7pXfy6kDkUVs=
 github.com/stretchr/testify v1.2.2/go.mod h1:a8OnRcib4nhh0OaRAV+Yts87kKdq0PP7pXfy6kDkUVs=
 github.com/stretchr/testify v1.3.0/go.mod h1:M5WIy9Dh21IEIfnGCwXGc5bZfKNJtfHm1UVUgZn+9EI=
```

M api/graph/api/generated.go => api/graph/api/generated.go +139 -4

```
@@ 50,10 50,11 @@ type ResolverRoot interface {
 }

 type DirectiveRoot struct {
-       Access    func(ctx context.Context, obj interface{}, next graphql.Resolver, scope model.AccessScope, kind
model.AccessKind) (res interface{}, err error)
-       Internal  func(ctx context.Context, obj interface{}, next graphql.Resolver) (res interface{}, err error)
-       Private   func(ctx context.Context, obj interface{}, next graphql.Resolver) (res interface{}, err error)
-       Scopehelp func(ctx context.Context, obj interface{}, next graphql.Resolver, details string) (res interface{},
err error)
+       Access       func(ctx context.Context, obj interface{}, next graphql.Resolver, scope model.AccessScope, kind
model.AccessKind) (res interface{}, err error)
+       Anoninternal func(ctx context.Context, obj interface{}, next graphql.Resolver) (res interface{}, err error)
+       Internal     func(ctx context.Context, obj interface{}, next graphql.Resolver) (res interface{}, err error)
+       Private      func(ctx context.Context, obj interface{}, next graphql.Resolver) (res interface{}, err error)
+       Scopehelp    func(ctx context.Context, obj interface{}, next graphql.Resolver, details string) (res
interface{}, err error)
 }

 type ComplexityRoot struct {


@@ 93,6 94,7 @@ type ComplexityRoot struct {
                IssueAuthorizationCode    func(childComplexity int, clientUUID string, grants string) int
                IssueOAuthGrant           func(childComplexity int, authorization string, clientSecret string) int
                IssuePersonalAccessToken  func(childComplexity int, grants *string, comment *string) int
+               RegisterAccount           func(childComplexity int, email string, username string, password string,
pgpKey *string, invite *string) int
                RegisterOAuthClient       func(childComplexity int, redirectURI string, clientName string,
clientDescription *string, clientURL *string) int
                RevokeOAuthClient         func(childComplexity int, uuid string) int
                RevokeOAuthGrant          func(childComplexity int, hash string) int


@@ 276,6 278,7 @@ type MutationResolver interface {
        UpdateSSHKey(ctx context.Context, id int) (*model.SSHKey, error)
        CreateWebhook(ctx context.Context, config model.ProfileWebhookInput) (model.WebhookSubscription, error)
        DeleteWebhook(ctx context.Context, id int) (model.WebhookSubscription, error)
+       RegisterAccount(ctx context.Context, email string, username string, password string, pgpKey *string, invite
*string) (*model.User, error)
        RegisterOAuthClient(ctx context.Context, redirectURI string, clientName string, clientDescription *string,
clientURL *string) (*model.OAuthClientRegistration, error)
        RevokeOAuthClient(ctx context.Context, uuid string) (*model.OAuthClient, error)
        RevokeOAuthGrant(ctx context.Context, hash string) (*model.OAuthGrant, error)


@@ 550,6 553,18 @@ func (e *executableSchema) Complexity(typeName, field string, childComplexity in

                return e.complexity.Mutation.IssuePersonalAccessToken(childComplexity, args["grants"].(*string),
args["comment"].(*string)), true

+       case "Mutation.registerAccount":
+               if e.complexity.Mutation.RegisterAccount == nil {
+                       break
+               }
+
+               args, err := ec.field_Mutation_registerAccount_args(context.TODO(), rawArgs)
+               if err != nil {
+                       return 0, false
+               }
+
+               return e.complexity.Mutation.RegisterAccount(childComplexity, args["email"].(string),
args["username"].(string), args["password"].(string), args["pgpKey"].(*string), args["invite"].(*string)), true
+
        case "Mutation.registerOAuthClient":
                if e.complexity.Mutation.RegisterOAuthClient == nil {
                        break


@@ 1513,6 1528,7 @@ directive @private on FIELD_DEFINITION
 # This used to decorate fields which are for internal use, and are not
```

```
 # available to normal API users.
 directive @internal on FIELD_DEFINITION
+directive @anoninternal on FIELD_DEFINITION

 # Used to provide a human-friendly description of an access scope.
 directive @scopehelp(details: String!) on ENUM_VALUE


@@ 1920,6 1936,13 @@ type Mutation {
   ### The following resolvers are for internal use. ###
   ###                                               ###

+  # Registers a new account.
+  registerAccount(email: String!,
+    username: String!,
+    password: String!,
+    pgpKey: String,
+    invite: String): User @anoninternal
+
   # Registers an OAuth client. Only OAuth 2.0 confidental clients are
   # supported.
   registerOAuthClient(


@@ 2160,6 2183,57 @@ func (ec *executionContext) field_Mutation_issuePersonalAccessToken_args(ctx con
        return args, nil
 }

+func (ec *executionContext) field_Mutation_registerAccount_args(ctx context.Context, rawArgs map[string]interface{})
(map[string]interface{}, error) {
+       var err error
+       args := map[string]interface{}{}
+       var arg0 string
+       if tmp, ok := rawArgs["email"]; ok {
+               ctx := graphql.WithPathContext(ctx, graphql.NewPathWithField("email"))
+               arg0, err = ec.unmarshalNString2string(ctx, tmp)
+               if err != nil {
+                       return nil, err
+               }
+       }
+       args["email"] = arg0
+       var arg1 string
+       if tmp, ok := rawArgs["username"]; ok {
+               ctx := graphql.WithPathContext(ctx, graphql.NewPathWithField("username"))
+               arg1, err = ec.unmarshalNString2string(ctx, tmp)
+               if err != nil {
+                       return nil, err
+               }
+       }
+       args["username"] = arg1
+       var arg2 string
+       if tmp, ok := rawArgs["password"]; ok {
+               ctx := graphql.WithPathContext(ctx, graphql.NewPathWithField("password"))
+               arg2, err = ec.unmarshalNString2string(ctx, tmp)
+               if err != nil {
+                       return nil, err
+               }
+       }
+       args["password"] = arg2
+       var arg3 *string
+       if tmp, ok := rawArgs["pgpKey"]; ok {
+               ctx := graphql.WithPathContext(ctx, graphql.NewPathWithField("pgpKey"))
+               arg3, err = ec.unmarshalOString2ᚖstring(ctx, tmp)
+               if err != nil {
+                       return nil, err
+               }
+       }
+       args["pgpKey"] = arg3
+       var arg4 *string
+       if tmp, ok := rawArgs["invite"]; ok {
+               ctx := graphql.WithPathContext(ctx, graphql.NewPathWithField("invite"))
+               arg4, err = ec.unmarshalOString2ᚖstring(ctx, tmp)
```

```
+               if err != nil {
+                       return nil, err
+               }
+       }
+       args["invite"] = arg4
+       return args, nil
+}
+
 func (ec *executionContext) field_Mutation_registerOAuthClient_args(ctx context.Context, rawArgs
map[string]interface{}) (map[string]interface{}, error) {
        var err error
        args := map[string]interface{}{}


@@ 3552,6 3626,65 @@ func (ec *executionContext) _Mutation_deleteWebhook(ctx context.Context, field g
        return ec.marshalOWebhookSubscription2git.sr.ht/~sircmpwn/meta.sr.ht/api/graph/model.WebhookSubscription(ctx,
field.Selections, res)
 }

+func (ec *executionContext) _Mutation_registerAccount(ctx context.Context, field graphql.CollectedField) (ret
graphql.Marshaler) {
+       defer func() {
+               if r := recover(); r != nil {
+                       ec.Error(ctx, ec.Recover(ctx, r))
+                       ret = graphql.Null
+               }
+       }()
+       fc := &graphql.FieldContext{
+               Object:     "Mutation",
+               Field:      field,
+               Args:       nil,
+               IsMethod:   true,
+               IsResolver: true,
+       }
+
+       ctx = graphql.WithFieldContext(ctx, fc)
+       rawArgs := field.ArgumentMap(ec.Variables)
+       args, err := ec.field_Mutation_registerAccount_args(ctx, rawArgs)
+       if err != nil {
+               ec.Error(ctx, err)
+               return graphql.Null
+       }
+       fc.Args = args
+       resTmp, err := ec.ResolverMiddleware(ctx, func(rctx context.Context) (interface{}, error) {
+               directive0 := func(rctx context.Context) (interface{}, error) {
+                       ctx = rctx // use context from middleware stack in children
+                       return ec.resolvers.Mutation().RegisterAccount(rctx, args["email"].(string),
args["username"].(string), args["password"].(string), args["pgpKey"].(*string), args["invite"].(*string))
+               }
+               directive1 := func(ctx context.Context) (interface{}, error) {
+                       if ec.directives.Anoninternal == nil {
+                               return nil, errors.New("directive anoninternal is not implemented")
+                       }
+                       return ec.directives.Anoninternal(ctx, nil, directive0)
+               }
+
+               tmp, err := directive1(rctx)
+               if err != nil {
+                       return nil, graphql.ErrorOnPath(ctx, err)
+               }
+               if tmp == nil {
+                       return nil, nil
+               }
+               if data, ok := tmp.(*model.User); ok {
+                       return data, nil
+               }
+               return nil, fmt.Errorf(`unexpected type %T from directive, should be
*git.sr.ht/~sircmpwn/meta.sr.ht/api/graph/model.User`, tmp)
+       })
+       if err != nil {
+               ec.Error(ctx, err)
+               return graphql.Null
```

```
+       }
+       if resTmp == nil {
+               return graphql.Null
+       }
+       res := resTmp.(*model.User)
+       fc.Result = res
+       return ec.marshalOUser2␀git␗sr␗ht⁄␘sircmpwn⁄meta␗sr␗ht⁄api⁄graph⁄model+User(ctx, field.Selections, res)
+}
+
 func (ec *executionContext) _Mutation_registerOAuthClient(ctx context.Context, field graphql.CollectedField) (ret
graphql.Marshaler) {
        defer func() {
                if r := recover(); r != nil {


@@ 9706,6 9839,8 @@ func (ec *executionContext) _Mutation(ctx context.Context, sel ast.SelectionSet)
                        }
                case "deleteWebhook":
                        out.Values[i] = ec._Mutation_deleteWebhook(ctx, field)
+               case "registerAccount":
+                       out.Values[i] = ec._Mutation_registerAccount(ctx, field)
                case "registerOAuthClient":
                        out.Values[i] = ec._Mutation_registerOAuthClient(ctx, field)
                        if out.Values[i] == graphql.Null {
```

A api/graph/blacklist.go => api/graph/blacklist.go +1219 -0

```
@@ 0,0 1,1219 @@
+package graph
+
+// https://gist.github.com/michenriksen/8710649
+// Keep sorted
+var emailBlacklist []string = []string{
+    "0815.ru",
+    "0815.ru",
+    "0hio0ak.com",
+    "0wnd.net",
+    "0wnd.org",
+    "10minutemail.co.za",
+    "10minutemail.com",
+    "123-m.com",
+    "1fsdfdsfsdf.tk",
+    "1mail.x24hr.com",
+    "1pad.de",
+    "1secmail.com",
+    "1secmail.net",
+    "1secmail.org",
+    "20minutemail.com",
+    "21cn.com",
+    "23.8.dnsabr.com",
+    "2fdgdfgdfgdf.tk",
+    "2prong.com",
+    "30minutemail.com",
+    "32core.live",
+    "33mail.com",
+    "3trtretgfrfe.tk",
+    "4dentalsolutions.com",
+    "4gfdsgfdgfd.tk",
+    "4warding.com",
+    "5ghgfhfghfgh.tk",
+    "6hjgjhgkilkj.tk",
+    "6paq.com",
+    "7tags.com",
+    "8.dnsabr.com",
+    "888.dnS-clouD.NET",
+    "99email.xyz",
+    "9ox.net",
+    "B.cr.cloUdnS.asia",
+    "BD.dns-cloud.net",
+    "CR.cloudns.asia",
+    "DVD.dns-cloud.net",
+    "DVD.dnsabr.com",
+    "Disposable.ml",
+    "MSFT.cloudns.asia",
+    "SHIT.dns-cloud.net",
+    "SHIT.dnsabr.com",
+    "TLS.cloudns.asia",
+    "YX.dns-cloud.net",
+    "a-bc.net",
+    "adult-work.info",
+    "agedmail.com",
+    "ama-trade.de",
+    "amilegit.com",
+    "amiri.net",
+    "amiriindustries.com",
+    "anonmails.de",
+    "anonymbox.com",
+    "anonymized.org",
+    "antichef.com",
+    "antichef.net",
+    "antireg.ru",
+    "antispam.de",
+    "antispammail.de",
+    "armyspy.com",
+    "artman-conception.com",
+    "asia.dnsabr.com",
+    "awdrt.net",
+    "azmeil.tk",
```

```
+    "badlion.co.uk",
+    "baxomale.ht.cx",
+    "beefmilk.com",
+    "bigstring.com",
+    "binkmail.com",
+    "bio-muesli.net",
+    "biyac.com",
+    "blackturtle.xyz",
+    "bobmail.info",
+    "bodhi.lawlita.com",
+    "bofthew.com",
+    "bootybay.de",
+    "boun.cr",
+    "bouncr.com",
+    "breakthru.com",
+    "brefmail.com",
+    "bsnow.net",
+    "bspamfree.org",
+    "btc.glass",
+    "budaya-tionghoa.com",
+    "budayationghoa.com",
+    "bugmenot.com",
+    "bund.us",
+    "burstmail.info",
+    "buymoreplays.com",
+    "byom.de",
+    "c2.hu",
+    "card.zp.ua",
+    "casualdx.com",
+    "cek.pm",
+    "centermail.com",
+    "centermail.net",
+    "chammy.info",
+    "chapedia.net",
+    "chapedia.org",
+    "chasefreedomactivate.com",
+    "childsavetrust.org",
+    "chitthi.in",
+    "chogmail.com",
+    "choicemail1.com",
+    "clixser.com",
+    "cmail.net",
+    "cmail.org",
+    "coldemail.info",
+    "cool.fr.nf",
+    "coolmailcool.com",
+    "corona.is.bullsht.dedyn.io",
+    "courriel.fr.nf",
+    "courrieltemporaire.com",
+    "cpmail.life",
+    "crapmail.org",
+    "cust.in",
+    "cuvox.de",
+    "d3p.dk",
+    "dacoolest.com",
+    "dandikmail.com",
+    "dayrep.com",
+    "dcemail.com",
+    "deadaddress.com",
+    "deadspam.com",
+    "delikkt.de",
+    "despam.it",
+    "despammed.com",
+    "devnullmail.com",
+    "dfgh.net",
+    "digitalsanctuary.com",
+    "dingbone.com",
+    "discard.email",
+    "discardmail.com",
+    "discardmail.de",
+    "disposable-email.ml",
+    "disposableaddress.com",
```

```
+    "disposableemailaddresses.com",
+    "disposableinbox.com",
+    "dispose.it",
+    "dispostable.com",
+    "dodgeit.com",
+    "dodgit.com",
+    "donemail.ru",
+    "dontreg.com",
+    "dontsendmespam.de",
+    "drdrb.net",
+    "dristypat.com",
+    "dump-email.info",
+    "dumpandjunk.com",
+    "dumpyemail.com",
+    "e-mail.com",
+    "e-mail.org",
+    "e4ward.com",
+    "easytrashmail.com",
+    "einmalmail.de",
+    "einrot.com",
+    "eintagsmail.de",
+    "emailgo.de",
+    "emailias.com",
+    "emaillime.com",
+    "emailsensei.com",
+    "emailtemporanea.com",
+    "emailtemporanea.net",
+    "emailtemporar.ro",
+    "emailtemporario.com.br",
+    "emailthe.net",
+    "emailtmp.com",
+    "emailwarden.com",
+    "emailx.at.hm",
+    "emailxfer.com",
+    "emeil.in",
+    "emeil.ir",
+    "emz.net",
+    "ero-tube.org",
+    "eu.dns-cloud.net",
+    "eu.dnsabr.com",
+    "evopo.com",
+    "explodemail.com",
+    "express.net.ua",
+    "eyepaste.com",
+    "fakeinbox.com",
+    "fakeinformation.com",
+    "fansworldwide.de",
+    "fantasymail.de",
+    "fexbos.ru",
+    "fexbox.org",
+    "fexpost.com",
+    "fightallspam.com",
+    "filzmail.com",
+    "fivemail.de",
+    "fleckens.hu",
+    "fouadps.cf",
+    "frapmail.com",
+    "freundin.ru",
+    "friendlymail.co.uk",
+    "from.onmypc.info",
+    "fshare.ootech.vn",
+    "fuckingduh.com",
+    "fudgerub.com",
+    "fyii.de",
+    "garliclife.com",
+    "gehensiemirnichtaufdensack.de",
+    "geneseeit.com",
+    "get2mail.fr",
+    "getairmail.com",
+    "getmails.eu",
+    "getonemail.com",
+    "giantmail.de",
```

```
+    "girlsundertheinfluence.com",
+    "gishpuppy.com",
+    "gmaile.design",
+    "gmial.com",
+    "goemailgo.com",
+    "gotmail.net",
+    "gotmail.org",
+    "gotti.otherinbox.com",
+    "great-host.in",
+    "greensloth.com",
+    "grr.la",
+    "gsrv.co.uk",
+    "guerillamail.biz",
+    "guerillamail.com",
+    "guerrillamail.biz",
+    "guerrillamail.com",
+    "guerrillamail.de",
+    "guerrillamail.info",
+    "guerrillamail.net",
+    "guerrillamail.org",
+    "guerrillamailblock.com",
+    "gustr.com",
+    "harakirimail.com",
+    "hat-geld.de",
+    "hatespam.org",
+    "herp.in",
+    "hidemail.de",
+    "hidemyass.fun",
+    "hidzz.com",
+    "historictheology.com",
+    "hmamail.com",
+    "hopemail.biz",
+    "hostux.ninja",
+    "ieh-mail.de",
+    "igosad.tech",
+    "ikbenspamvrij.nl",
+    "imails.info",
+    "inbax.tk",
+    "inbox.si",
+    "inboxalias.com",
+    "inboxclean.com",
+    "inboxclean.org",
+    "infocom.zp.ua",
+    "inpwa.com",
+    "instant-mail.de",
+    "intopwa.com",
+    "intopwa.net",
+    "intopwa.org",
+    "ip6.li",
+    "irish2me.com",
+    "iwi.net",
+    "jetable.com",
+    "jetable.fr.nf",
+    "jetable.net",
+    "jetable.org",
+    "jnxjn.com",
+    "jourrapide.com",
+    "jsrsolutions.com",
+    "kaaaxcreators.tk",
+    "kasmail.com",
+    "kaspop.com",
+    "ketoblazepro.com",
+    "killmail.com",
+    "killmail.net",
+    "kittenemail.xyz",
+    "klassmaster.com",
+    "klzlk.com",
+    "knol-power.nl",
+    "kost.party",
+    "koszmail.pl",
+    "kurzepost.de",
+    "lajoska.pe.hu",
```

```
+    "lawlita.com",
+    "letthemeatspam.com",
+    "lhsdv.com",
+    "lifebyfood.com",
+    "link2mail.net",
+    "litedrop.com",
+    "lol.ovpn.to",
+    "lolfreak.net",
+    "lookugly.com",
+    "lortemail.dk",
+    "lr78.com",
+    "lroid.com",
+    "lukop.dk",
+    "m.cloudns.cl",
+    "m21.cc",
+    "maa.567map.xyz",
+    "mail-filter.com",
+    "mail-temporaire.fr",
+    "mail.by",
+    "mail.igosad.me",
+    "mail.kaaaxcreators.tk",
+    "mail.mezimages.net",
+    "mail.mrgamin.ml",
+    "mail.zp.ua",
+    "mail1a.de",
+    "mail21.cc",
+    "mail2rss.org",
+    "mail333.com",
+    "mailbidon.com",
+    "mailbiz.biz",
+    "mailblocks.com",
+    "mailbox.in.ua",
+    "mailbucket.org",
+    "mailcat.biz",
+    "mailcatch.com",
+    "mailde.de",
+    "mailde.info",
+    "maildrop.cc",
+    "maileimer.de",
+    "mailexpire.com",
+    "mailfa.tk",
+    "mailforspam.com",
+    "mailfreeonline.com",
+    "mailg.ml",
+    "mailguard.me",
+    "mailin8r.com",
+    "mailinater.com",
+    "mailinator.com",
+    "mailinator.net",
+    "mailinator.org",
+    "mailinator2.com",
+    "mailincubator.com",
+    "mailismagic.com",
+    "mailme.lv",
+    "mailme24.com",
+    "mailmetrash.com",
+    "mailmoat.com",
+    "mailms.com",
+    "mailnesia.com",
+    "mailnull.com",
+    "mailorg.org",
+    "mailpick.biz",
+    "mailrock.biz",
+    "mailscrap.com",
+    "mailshell.com",
+    "mailsiphon.com",
+    "mailtemp.info",
+    "mailto.plus",
+    "mailtome.de",
+    "mailtothis.com",
+    "mailtrash.net",
+    "mailtv.net",
```

```
+    "mailtv.tv",
+    "mailzilla.com",
+    "makemetheking.com",
+    "manybrain.com",
+    "mbx.cc",
+    "meantinc.com",
+    "media.motornation.buzz",
+    "mega.zik.dj",
+    "meinspamschutz.de",
+    "meltmail.com",
+    "messagebeamer.de",
+    "mezimages.net",
+    "ministry-of-silly-walks.de",
+    "mintemail.com",
+    "misterpinball.de",
+    "miucce.com",
+    "mm.8.dnsabr.com",
+    "moncourrier.fr.nf",
+    "monemail.fr.nf",
+    "monmail.fr.nf",
+    "monumentmail.com",
+    "mowgli.jungleheart.com",
+    "mrdeeps.ml",
+    "mrgamin.cf",
+    "mrgamin.gq",
+    "mrgamin.ml",
+    "mt2009.com",
+    "mt2014.com",
+    "mycard.net.ua",
+    "mycleaninbox.net",
+    "mymail-in.net",
+    "mypacks.net",
+    "mypartyclip.de",
+    "myphantomemail.com",
+    "mysamp.de",
+    "mytempemail.com",
+    "mytempmail.com",
+    "mytrashmail.com",
+    "nabuma.com",
+    "neomailbox.com",
+    "nepwk.com",
+    "nervmich.net",
+    "nervtmich.net",
+    "netmails.com",
+    "netmails.net",
+    "neverbox.com",
+    "nice-4u.com",
+    "nincsmail.hu",
+    "nnh.com",
+    "no-spam.ws",
+    "noblepioneer.com",
+    "nomail.pw",
+    "nomail.xl.cx",
+    "nomail2me.com",
+    "nomorespamemails.com",
+    "nospam.ze.tc",
+    "nospam4.us",
+    "nospamfor.us",
+    "nospammail.net",
+    "notmailinator.com",
+    "notmyemail.tech",
+    "now.mefound.com",
+    "nowhere.org",
+    "nowmymail.com",
+    "nucleant.org",
+    "nurfuerspam.de",
+    "nus.edu.sg",
+    "objectmail.com",
+    "obobbo.com",
+    "odnorazovoe.ru",
+    "ondemandemail.top",
+    "oneoffemail.com",
```

```
+    "onewaymail.com",
+    "onlatedotcom.info",
+    "online.ms",
+    "opayq.com",
+    "ordinaryamerican.net",
+    "otherinbox.com",
+    "ovpn.to",
+    "owlpic.com",
+    "pancakemail.com",
+    "pcusers.otherinbox.com",
+    "pecinan.com",
+    "pecinan.net",
+    "pecinan.org",
+    "pflege-schoene-haut.de",
+    "pjjkp.com",
+    "plexolan.de",
+    "poczta.onet.pl",
+    "politikerclub.de",
+    "poofy.org",
+    "pookmail.com",
+    "postheo.de",
+    "powerencry.com",
+    "privacy.net",
+    "privatdemail.net",
+    "proxymail.eu",
+    "prtnx.com",
+    "putthisinyourspamdatabase.com",
+    "putthisinyourspamdatabase.com",
+    "pw.8.dnsabr.com",
+    "pw.epac.to",
+    "qq.com",
+    "qq.com",
+    "quickinbox.com",
+    "rcpt.at",
+    "reallymymail.com",
+    "realtyalerts.ca",
+    "recode.me",
+    "recursor.net",
+    "relay.firefox.com",
+    "reliable-mail.com",
+    "rhyta.com",
+    "rmqkr.net",
+    "rover.info",
+    "royal.net",
+    "rtrtr.com",
+    "s0ny.net",
+    "s0ny.net",
+    "safe-mail.net",
+    "safeemail.xyz",
+    "safersignup.de",
+    "safetymail.info",
+    "safetypost.de",
+    "saynotospams.com",
+    "schafmail.de",
+    "schrott-email.de",
+    "secretemail.de",
+    "secure-mail.biz",
+    "senseless-entertainment.com",
+    "services391.com",
+    "sexy.camdvr.org",
+    "sharklasers.com",
+    "shieldemail.com",
+    "shiftmail.com",
+    "shitmail.me",
+    "shitware.nl",
+    "shmeriously.com",
+    "shortmail.net",
+    "sibmail.com",
+    "sinnlos-mail.de",
+    "slapsfromlastnight.com",
+    "slaskpost.se",
+    "smack.email",
```

```
+    "smashmail.de",
+    "smashmail.de",
+    "smellfear.com",
+    "snakemail.com",
+    "sneakemail.com",
+    "sneakmail.de",
+    "snkmail.com",
+    "sofimail.com",
+    "sogetthis.com",
+    "solpatu.space",
+    "solvemail.info",
+    "soodonims.com",
+    "spam4.me",
+    "spamail.de",
+    "spamarrest.com",
+    "spambob.net",
+    "spambog.com",
+    "spambog.de",
+    "spambog.ru",
+    "spambog.ru",
+    "spambox.us",
+    "spamcannon.com",
+    "spamcannon.net",
+    "spamcon.org",
+    "spamcorptastic.com",
+    "spamcowboy.com",
+    "spamcowboy.net",
+    "spamcowboy.org",
+    "spamday.com",
+    "spamex.com",
+    "spamfree.eu",
+    "spamfree24.com",
+    "spamfree24.de",
+    "spamfree24.org",
+    "spamgoes.in",
+    "spamgourmet.com",
+    "spamgourmet.net",
+    "spamgourmet.org",
+    "spamherelots.com",
+    "spamherelots.com",
+    "spamhereplease.com",
+    "spamhereplease.com",
+    "spamhole.com",
+    "spamify.com",
+    "spaml.de",
+    "spammotel.com",
+    "spamobox.com",
+    "spamslicer.com",
+    "spamspot.com",
+    "spamthis.co.uk",
+    "spamtroll.net",
+    "speed.1s.fr",
+    "speedfocus.biz",
+    "spoofmail.de",
+    "ssl.tls.cloudns.ASIA",
+    "stuffmail.de",
+    "super-auswahl.de",
+    "supergreatmail.com",
+    "supermailer.jp",
+    "superrito.com",
+    "superstachel.de",
+    "suremail.info",
+    "sweetxxx.de",
+    "t.woeishyang.com",
+    "talkinator.com",
+    "techwizardent.me",
+    "teewars.org",
+    "teleworm.com",
+    "teleworm.us",
+    "temp-mail.org",
+    "temp-mail.ru",
+    "tempe-mail.com",
```

```
+    "tempemail.co.za",
+    "tempemail.com",
+    "tempemail.info",
+    "tempemail.net",
+    "tempemail.net",
+    "tempes.gq",
+    "tempinbox.co.uk",
+    "tempinbox.com",
+    "tempmail.eu",
+    "tempmail.wizardmail.tech",
+    "tempmaildemo.com",
+    "tempmailer.com",
+    "tempmailer.de",
+    "tempomail.fr",
+    "temporary-mail.net",
+    "temporaryemail.net",
+    "temporaryforwarding.com",
+    "temporaryinbox.com",
+    "temporarymailaddress.com",
+    "tempr.email",
+    "tempthe.net",
+    "thankyou2010.com",
+    "thc.st",
+    "thelimestones.com",
+    "thisisnotmyrealemail.com",
+    "thismail.net",
+    "throwawayemailaddress.com",
+    "tilien.com",
+    "tittbit.in",
+    "tizi.com",
+    "tmailinator.com",
+    "tokyoto.site",
+    "toomail.biz",
+    "topranklist.de",
+    "tradermail.info",
+    "trap-mail.de",
+    "trash-mail.at",
+    "trash-mail.com",
+    "trash-mail.de",
+    "trash2009.com",
+    "trashdevil.com",
+    "trashemail.de",
+    "trashmail.at",
+    "trashmail.com",
+    "trashmail.de",
+    "trashmail.me",
+    "trashmail.net",
+    "trashmail.org",
+    "trashymail.com",
+    "trialmail.de",
+    "trillianpro.com",
+    "truthfinderlogin.com",
+    "twinmail.de",
+    "twitter-sign-in.cf",
+    "tyldd.com",
+    "uggsrock.com",
+    "umail.net",
+    "upived.o",
+    "uroid.com",
+    "us.af",
+    "venompen.com",
+    "verygrealemail.com",
+    "viditag.com",
+    "viralplays.com",
+    "virtual-generations.com",
+    "vpn.st",
+    "vsimcard.com",
+    "vubby.com",
+    "wasteland.rfc822.org",
+    "webemail.me",
+    "weg-werf-email.de",
+    "wegwerf-emails.de",
```

```
+    "wegwerfadresse.de",
+    "wegwerfemail.com",
+    "wegwerfemail.de",
+    "wegwerfmail.de",
+    "wegwerfmail.info",
+    "wegwerfmail.net",
+    "wegwerfmail.org",
+    "wellsfargocomcardholders.com",
+    "wh4f.org",
+    "whyspam.me",
+    "willhackforfood.biz",
+    "willselfdestruct.com",
+    "winemaven.info",
+    "wronghead.com",
+    "www.e4ward.com",
+    "www.mailinator.com",
+    "wwwnew.eu",
+    "x.ip6.li",
+    "xagloo.com",
+    "xemaps.com",
+    "xents.com",
+    "xmaily.com",
+    "xoxy.net",
+    "yep.it",
+    "yogamaven.com",
+    "yopmail.com",
+    "yopmail.fr",
+    "yopmail.net",
+    "you.has.dating",
+    "yourdomain.com",
+    "yuurok.com",
+    "z1p.biz",
+    "za.com",
+    "zehnminuten.de",
+    "zehnminutenmail.de",
+    "zippymail.info",
+    "zoemail.net",
+    "zomg.info",
+}
+
+// https://github.com/marteinn/The-Big-Username-Blacklist
+// Keep sorted
+var usernameBlacklist []string = []string{
+    ".htaccess",
+    ".htpasswd",
+    ".well_known",
+    "400",
+    "401",
+    "403",
+    "404",
+    "405",
+    "406",
+    "407",
+    "408",
+    "409",
+    "410",
+    "411",
+    "412",
+    "413",
+    "414",
+    "415",
+    "416",
+    "417",
+    "421",
+    "422",
+    "423",
+    "424",
+    "426",
+    "428",
+    "429",
+    "431",
+    "500",
```

```
+    "501",
+    "502",
+    "503",
+    "504",
+    "505",
+    "506",
+    "507",
+    "508",
+    "509",
+    "510",
+    "511",
+    "about",
+    "about_us",
+    "abuse",
+    "access",
+    "account",
+    "accounts",
+    "ad",
+    "add",
+    "admin",
+    "administration",
+    "administrator",
+    "ads",
+    "advertise",
+    "advertising",
+    "aes128_ctr",
+    "aes128_gcm",
+    "aes192_ctr",
+    "aes256_ctr",
+    "aes256_gcm",
+    "affiliate",
+    "affiliates",
+    "ajax",
+    "alert",
+    "alerts",
+    "alpha",
+    "amp",
+    "analytics",
+    "api",
+    "app",
+    "apps",
+    "asc",
+    "assets",
+    "atom",
+    "auth",
+    "authentication",
+    "authorize",
+    "autoconfig",
+    "autodiscover",
+    "avatar",
+    "backup",
+    "banner",
+    "banners",
+    "beta",
+    "billing",
+    "billings",
+    "blog",
+    "blogs",
+    "board",
+    "bookmark",
+    "bookmarks",
+    "broadcasthost",
+    "business",
+    "buy",
+    "cache",
+    "calendar",
+    "campaign",
+    "captcha",
+    "careers",
+    "cart",
+    "cas",
+    "categories",
```

```
+    "category",
+    "cdn",
+    "cgi",
+    "cgi_bin",
+    "chacha20_poly1305",
+    "change",
+    "channel",
+    "channels",
+    "chart",
+    "chat",
+    "checkout",
+    "clear",
+    "client",
+    "close",
+    "cms",
+    "com",
+    "comment",
+    "comments",
+    "community",
+    "compare",
+    "compose",
+    "config",
+    "connect",
+    "contact",
+    "contest",
+    "cookies",
+    "copy",
+    "copyright",
+    "count",
+    "create",
+    "crossdomain.xml",
+    "css",
+    "curve25519_sha256",
+    "customer",
+    "customers",
+    "customize",
+    "dashboard",
+    "db",
+    "ddevault",
+    "deals",
+    "debug",
+    "delete",
+    "desc",
+    "dev",
+    "developer",
+    "developers",
+    "diffie_hellman_group14_sha1",
+    "diffie_hellman_group_exchange_sha256",
+    "disconnect",
+    "discuss",
+    "dns",
+    "dns0",
+    "dns1",
+    "dns2",
+    "dns3",
+    "dns4",
+    "docs",
+    "documentation",
+    "domain",
+    "download",
+    "downloads",
+    "downvote",
+    "draft",
+    "drop",
+    "ecdh_sha2_nistp256",
+    "ecdh_sha2_nistp384",
+    "ecdh_sha2_nistp521",
+    "edit",
+    "editor",
+    "email",
+    "enterprise",
+    "error",
```

```
+    "errors",
+    "event",
+    "events",
+    "example",
+    "exception",
+    "exit",
+    "explore",
+    "export",
+    "extensions",
+    "false",
+    "family",
+    "faq",
+    "faqs",
+    "favicon.ico",
+    "features",
+    "feed",
+    "feedback",
+    "feeds",
+    "file",
+    "files",
+    "filter",
+    "follow",
+    "follower",
+    "followers",
+    "following",
+    "fonts",
+    "forgot",
+    "forgot_password",
+    "forgotpassword",
+    "form",
+    "forms",
+    "forum",
+    "forums",
+    "friend",
+    "friends",
+    "ftp",
+    "get",
+    "git",
+    "go",
+    "group",
+    "groups",
+    "guest",
+    "guidelines",
+    "guides",
+    "head",
+    "header",
+    "help",
+    "hide",
+    "hmac_sha",
+    "hmac_sha1",
+    "hmac_sha1_etm",
+    "hmac_sha2_256",
+    "hmac_sha2_256_etm",
+    "hmac_sha2_512",
+    "hmac_sha2_512_etm",
+    "home",
+    "host",
+    "hosting",
+    "hostmaster",
+    "htpasswd",
+    "http",
+    "httpd",
+    "https",
+    "humans.txt",
+    "icons",
+    "images",
+    "imap",
+    "img",
+    "import",
+    "info",
+    "insert",
+    "investors",
```

```
+    "invitations",
+    "invite",
+    "invites",
+    "invoice",
+    "is",
+    "isatap",
+    "issues",
+    "it",
+    "jobs",
+    "join",
+    "js",
+    "json",
+    "keybase.txt",
+    "learn",
+    "legal",
+    "license",
+    "licensing",
+    "limit",
+    "live",
+    "load",
+    "local",
+    "localdomain",
+    "localhost",
+    "lock",
+    "login",
+    "logout",
+    "lost_password",
+    "mail",
+    "mail0",
+    "mail1",
+    "mail2",
+    "mail3",
+    "mail4",
+    "mail5",
+    "mail6",
+    "mail7",
+    "mail8",
+    "mail9",
+    "mailer_daemon",
+    "mailerdaemon",
+    "map",
+    "marketing",
+    "marketplace",
+    "master",
+    "me",
+    "media",
+    "member",
+    "members",
+    "message",
+    "messages",
+    "metrics",
+    "mis",
+    "mobile",
+    "moderator",
+    "modify",
+    "more",
+    "mx",
+    "my",
+    "net",
+    "network",
+    "new",
+    "news",
+    "newsletter",
+    "newsletters",
+    "next",
+    "nil",
+    "no_reply",
+    "nobody",
+    "noc",
+    "none",
+    "noreply",
+    "notification",
```

```
+    "notifications",
+    "ns",
+    "ns0",
+    "ns1",
+    "ns2",
+    "ns3",
+    "ns4",
+    "ns5",
+    "ns6",
+    "ns7",
+    "ns8",
+    "ns9",
+    "null",
+    "oauth",
+    "oauth2",
+    "offer",
+    "offers",
+    "online",
+    "openid",
+    "order",
+    "orders",
+    "overview",
+    "owner",
+    "page",
+    "pages",
+    "partners",
+    "passwd",
+    "password",
+    "pay",
+    "payment",
+    "payments",
+    "photo",
+    "photos",
+    "pixel",
+    "plans",
+    "plugins",
+    "policies",
+    "policy",
+    "pop",
+    "pop3",
+    "popular",
+    "portfolio",
+    "post",
+    "postfix",
+    "postmaster",
+    "poweruser",
+    "preferences",
+    "premium",
+    "press",
+    "previous",
+    "pricing",
+    "print",
+    "privacy",
+    "privacy_policy",
+    "private",
+    "prod",
+    "product",
+    "production",
+    "profile",
+    "profiles",
+    "project",
+    "projects",
+    "public",
+    "purchase",
+    "put",
+    "quota",
+    "redirect",
+    "reduce",
+    "refund",
+    "refunds",
+    "register",
+    "registration",
```

```
+    "remove",
+    "replies",
+    "reply",
+    "report",
+    "request",
+    "request_password",
+    "reset",
+    "reset_password",
+    "response",
+    "return",
+    "returns",
+    "review",
+    "reviews",
+    "robots.txt",
+    "root",
+    "rootuser",
+    "rsa_sha2_2",
+    "rsa_sha2_512",
+    "rss",
+    "rules",
+    "sales",
+    "save",
+    "script",
+    "sdk",
+    "search",
+    "secure",
+    "security",
+    "select",
+    "services",
+    "session",
+    "sessions",
+    "settings",
+    "setup",
+    "share",
+    "shift",
+    "shop",
+    "signin",
+    "signup",
+    "sircmpwn",
+    "sirhat",
+    "sirhit",
+    "site",
+    "sitemap",
+    "sites",
+    "smtp",
+    "sort",
+    "source",
+    "sourcehut",
+    "sql",
+    "srcht",
+    "srchut",
+    "srht",
+    "ssh",
+    "ssh_rsa",
+    "ssl",
+    "ssladmin",
+    "ssladministrator",
+    "sslwebmaster",
+    "stage",
+    "staging",
+    "stat",
+    "static",
+    "statistics",
+    "stats",
+    "status",
+    "store",
+    "style",
+    "styles",
+    "stylesheet",
+    "stylesheets",
+    "subdomain",
+    "subscribe",
```

```
+    "sudo",
+    "super",
+    "superuser",
+    "support",
+    "survey",
+    "sync",
+    "sysadmin",
+    "system",
+    "tablet",
+    "tag",
+    "tags",
+    "team",
+    "telnet",
+    "terms",
+    "terms_of_use",
+    "test",
+    "testimonials",
+    "theme",
+    "themes",
+    "today",
+    "tools",
+    "topic",
+    "topics",
+    "tour",
+    "training",
+    "translate",
+    "translations",
+    "trending",
+    "trial",
+    "true",
+    "umac_128",
+    "umac_128_etm",
+    "umac_64",
+    "umac_64_etm",
+    "undefined",
+    "unfollow",
+    "unsubscribe",
+    "update",
+    "upgrade",
+    "usenet",
+    "user",
+    "username",
+    "users",
+    "uucp",
+    "var",
+    "verify",
+    "video",
+    "view",
+    "void",
+    "vote",
+    "webmail",
+    "webmaster",
+    "website",
+    "widget",
+    "widgets",
+    "wiki",
+    "wpad",
+    "write",
+    "www",
+    "www1",
+    "www2",
+    "www3",
+    "www4",
+    "www_data",
+    "you",
+    "yourname",
+    "yourusername",
+    "zlib",
+}
```

M api/graph/resolver.go => api/graph/resolver.go +72 -11

```
@@ 6,6 6,7 @@ import (
        "fmt"
        "log"
        "net"
+       "regexp"
        "strings"
        "text/template"



@@ 23,6 24,10 @@ import (

 //go:generate go run github.com/99designs/gqlgen

+var (
+       usernameRE = regexp.MustCompile(`^[a-z_][a-z0-9_-]+$`)
+)
+
 type Resolver struct{}

 type AuthorizationPayload struct {


@@ 50,10 55,7 @@ func filterWebhooks(ctx context.Context) (sq.Sqlizer, error) {

 // Records an event in the authorized user's audit log.
 func recordAuditLog(ctx context.Context, eventType, details string) {
-       user := auth.ForContext(ctx)
-
-       var id int
-       if err := database.WithTx(ctx, nil, func(tx *sql.Tx) error {
+       database.WithTx(ctx, nil, func(tx *sql.Tx) error {
                var err error
                addr := server.RemoteAddr(ctx)
                if strings.ContainsRune(addr, ':') {


@@ 63,25 65,84 @@ func recordAuditLog(ctx context.Context, eventType, details string) {
                        }
                }

-               row := tx.QueryRowContext(ctx, `
+               user := auth.ForContext(ctx)
+               _, err = tx.ExecContext(ctx, `
                        INSERT INTO audit_log_entry (
                                created, user_id, ip_address, event_type, details
                        ) VALUES (
                                NOW() at time zone 'utc',
                                $1, $2, $3, $4
-                       ) RETURNING id;
+                       );
                `, user.UserID, addr, eventType, details)
-
-               if err := row.Scan(&id); err != nil {
-                       return err
+               if err != nil {
+                       panic(err)
                }

+               log.Printf("Audit log: %s: %s", eventType, details)
                return nil
-       }); err != nil {
+       })
+}
+
+func sendRegistrationConfirmation(ctx context.Context,
+       user *model.User, pgpKey *string, confirmation string) {
+       conf := config.ForContext(ctx)
+       siteName, ok := conf.Get("sr.ht", "site-name")
+       if !ok {
+               panic(fmt.Errorf("Expected [sr.ht]site-name in config"))
```

```
+	}
+	ownerName, ok := conf.Get("sr.ht", "owner-name")
+	if !ok {
+		panic(fmt.Errorf("Expected [sr.ht]owner-name in config"))
+	}
+
+	var header mail.Header
+	header.SetAddressList("To", []*mail.Address{
+		&mail.Address{"~" + user.Username, user.Email},
+	})
+	header.SetSubject(fmt.Sprintf("Confirm your %s registration", siteName))
+
+	type TemplateContext struct {
+		OwnerName    string
+		SiteName     string
+		Username     string
+		Root         string
+		Confirmation string
+	}
+	tctx := TemplateContext{
+		OwnerName:    ownerName,
+		SiteName:     siteName,
+		Username:     user.Username,
+		Root:         config.GetOrigin(conf, "meta.sr.ht", true),
+		Confirmation: confirmation,
+	}
+
+	tmpl := template.Must(template.New("security-event").Parse(`Hello ~{{.Username}}!
+
+You (or someone pretending to be you) have registered for an account on
+{{.SiteName}}.
+
+To complete your registration, please follow this link:
+
+{{.Root}}/confirm-account/{{.Confirmation}}
+
+If not, just ignore this email. If you have any questions, please reply
+to this email.
+
+--
+{{.OwnerName}}
+{{.SiteName}}`))
+
+	var body strings.Builder
+	err := tmpl.Execute(&body, tctx)
+	if err != nil {
+		panic(err)
+	}

-	log.Printf("Audit log (%d): %s: %s", id, eventType, details)
+	err = email.EnqueueStd(ctx, header,
+		strings.NewReader(body.String()), pgpKey)
+	if err != nil {
+		panic(err)
+	}
 }

 // Sends a security-related notice to the authorized user.
```

M api/graph/schema.graphqls => api/graph/schema.graphqls +8 -0

```
@@ 10,6 10,7 @@ directive @private on FIELD_DEFINITION
 # This used to decorate fields which are for internal use, and are not
 # available to normal API users.
 directive @internal on FIELD_DEFINITION
+directive @anoninternal on FIELD_DEFINITION

 # Used to provide a human-friendly description of an access scope.
 directive @scopehelp(details: String!) on ENUM_VALUE


@@ 417,6 418,13 @@ type Mutation {
   ### The following resolvers are for internal use. ###
   ###                                              ###

+  # Registers a new account.
+  registerAccount(email: String!,
+    username: String!,
+    password: String!,
+    pgpKey: String,
+    invite: String): User @anoninternal
+
   # Registers an OAuth client. Only OAuth 2.0 confidental clients are
   # supported.
   registerOAuthClient(
```

M api/graph/schema.resolvers.go => api/graph/schema.resolvers.go +218 -0

```
@@ 12,8 12,12 @@ import (
        "encoding/base64"
        "encoding/hex"
        "encoding/json"
+       "errors"
        "fmt"
+       "net"
        "net/url"
+       "sort"
+       "strconv"
        "strings"
        "time"



@@ 32,7 36,10 @@ import (
        sq "github.com/Masterminds/squirrel"
        "github.com/google/uuid"
        "github.com/lib/pq"
+       zxcvbn "github.com/nbutton23/zxcvbn-go"
+       "golang.org/x/crypto/bcrypt"
        "golang.org/x/crypto/openpgp"
+       "golang.org/x/crypto/openpgp/packet"
        "golang.org/x/crypto/ssh"
 )



@@ 167,6 174,8 @@ func (r *mutationResolver) UpdateUser(ctx context.Context, input map[string]inte
 }

 func (r *mutationResolver) CreatePGPKey(ctx context.Context, key string) (*model.PGPKey, error) {
+       // Note: You may also need to update the RegisterAccount resolver if you
+       // are working with this code.
        valid := valid.New(ctx)
        keys, err := openpgp.ReadArmoredKeyRing(strings.NewReader(key))
        valid.



@@ 549,6 558,215 @@ func (r *mutationResolver) DeleteWebhook(ctx context.Context, id int) (model.Web
        return &sub, nil
 }

+func (r *mutationResolver) RegisterAccount(ctx context.Context, email string, username string, password string,
pgpKey *string, invite *string) (*model.User, error) {
+       // Note: this resolver is used with anonymous internal auth, so most of the
+       // fields in auth.ForContext(ctx) are invalid.
+       valid := valid.New(ctx)
+       valid.Expect(len(username) >= 2 && len(username) <= 30,
+               "Username must be between 2 and 30 characters in length.").
+               WithField("username")
+       valid.Expect(usernameRE.MatchString(username),
+               "Username must use only lowercase letters, digits, underscores, and dashes, and must start with a
letter or underscore.").
+               WithField("username")
+       blacklist := sort.SearchStrings(usernameBlacklist, username)
+       valid.Expect(blacklist < len(usernameBlacklist) &&
+               usernameBlacklist[blacklist] != username,
+               "This username is not available").
+               WithField("username")
+
+       valid.Expect(len(email) <= 256,
+               "Email cannot be greater than 256 characters in length.").
+               WithField("email")
+       valid.Expect(strings.ContainsRune(email, '@'),
+               "This is not a valid email address.").
+               WithField("email")
+       parts := strings.Split(email, "@")
+       if len(parts) == 2 {
+               blacklist := sort.SearchStrings(emailBlacklist, strings.ToLower(parts[1]))
+               valid.Expect(blacklist < len(emailBlacklist) &&
```

```
+                       emailBlacklist[blacklist] != email,
+                       "Accounts are not permitted to use this email provider.").
+                       WithField("email")
+       }
+
+       valid.Expect(len(password) <= 512,
+               "Password must be no more than 512 characters in length.").
+               WithField("password")
+       conf := config.ForContext(ctx)
+       env, ok := conf.Get("sr.ht", "environment")
+       if ok && env == "production" {
+               strength := zxcvbn.PasswordStrength(password, []string{
+                       username,
+                       email,
+                       "sourcehut",
+                       "sr.ht",
+               })
+               valid.Expect(strength.Score >= 3,
+                       "This password is too weak. Longer passwords are better than complicated passwords. The use
of a password manager is strongly recommended.").
+                       WithField("password")
+       }
+
+       var pkey *packet.PublicKey
+       if pgpKey != nil {
+               // Note: You may also need to update the CreatePGPKey resolver if you
+               // are working with this code.
+               keys, err := openpgp.ReadArmoredKeyRing(strings.NewReader(*pgpKey))
+               valid.
+                       Expect(err == nil, "Invalid PGP key format: %v", err).
+                       WithField("key").
+                       And(len(keys) == 1, "Expected one key, found %d", len(keys)).
+                       WithField("key")
+               if !valid.Ok() {
+                       return nil, nil
+               }
+
+               entity := keys[0]
+               valid.Expect(entity.PrivateKey == nil, "There's a private key in here, yikes!")
+
+               pkey = entity.PrimaryKey
+               valid.Expect(pkey != nil && pkey.CanSign(),
+                       "No public keys suitable for signing found.")
+       }
+
+       if !valid.Ok() {
+               return nil, nil
+       }
+
+       invites := 0
+       inv, ok := conf.Get("meta.sr.ht::settings", "user-invites")
+       if ok {
+               var err error
+               invites, err = strconv.Atoi(inv)
+               if err != nil {
+                       panic(err)
+               }
+       }
+
+       pwhash, err := bcrypt.GenerateFromPassword(
+               []byte(password), bcrypt.DefaultCost)
+       if err != nil {
+               panic(err)
+       }
+
+       var seed [18]byte
+       if _, err := rand.Read(seed[:]); err != nil {
+               panic(err)
+       }
+       confirmation := base64.URLEncoding.EncodeToString(seed[:])
+
+       var user model.User
```

```
+	if err := database.WithTx(ctx, nil, func(tx *sql.Tx) error {
+		row := tx.QueryRowContext(ctx, `
+			INSERT INTO "user" (
+				created, updated, username, email, user_type, password,
+				confirmation_hash, invites
+			) VALUES (
+				NOW() at time zone 'utc',
+				NOW() at time zone 'utc',
+				$1, $2, 'unconfirmed', $3, $4, $5
+			)
+			RETURNING id, created, updated, username, email, user_type;
+		`, username, email, string(pwhash), confirmation, invites)
+
+		if err := row.Scan(&user.ID, &user.Created, &user.Updated,
+			&user.Username, &user.Email, &user.UserTypeRaw); err != nil {
+			if err, ok := err.(*pq.Error); ok &&
+				err.Code == "23505" && // unique_violation
+				err.Constraint == "ix_user_username" {
+				valid.Error("This username is already in use.").
+					WithField("username")
+				return errors.New("placeholder") // To rollback the transaction
+			}
+			if err, ok := err.(*pq.Error); ok &&
+				err.Code == "23505" && // unique_violation
+				err.Constraint == "user_email_unique" {
+				valid.Error("This email address is already in use.").
+					WithField("email")
+				return errors.New("placeholder") // To rollback the transaction
+			}
+			return err
+		}
+
+		if invite != nil {
+			row = tx.QueryRowContext(ctx, `
+				UPDATE invite
+				SET recipient_id = $1
+				WHERE invite_hash = $2 AND recipient_id IS NULL
+				RETURNING id;
+			`, user.ID, *invite)
+
+			var id int
+			if err := row.Scan(&id); err != nil {
+				if err == sql.ErrNoRows {
+					valid.Error("The invite code you've used is invalid or expired.").
+						WithField("invite")
+					return errors.New("placeholder")
+				}
+				return err
+			}
+		}
+
+		addr := server.RemoteAddr(ctx)
+		if strings.ContainsRune(addr, ':') {
+			addr, _, err = net.SplitHostPort(addr)
+			if err != nil {
+				panic(err)
+			}
+		}
+
+		_, err = tx.ExecContext(ctx, `
+			INSERT INTO audit_log_entry (
+				created, user_id, ip_address, event_type, details
+			) VALUES (
+				NOW() at time zone 'utc',
+				$1, $2, $3, $4
+			);`, user.ID, addr,
+			"account registered",
+			fmt.Sprintf("registered ~%s (%s)", user.Username, user.Email))
+		if err != nil {
+			panic(err)
+		}
+
```

```
+               if pkey != nil {
+                       row = tx.QueryRowContext(ctx, `
+                                       INSERT INTO pgpkey (
+                                               created, user_id, key, fingerprint
+                                       ) VALUES (
+                                               NOW() at time zone 'utc',
+                                               $1, $2, $3
+                                       ) RETURNING id;
+                               `, user.ID, *pgpKey, pkey.Fingerprint[:])
+                       var id int
+                       if err := row.Scan(&id); err != nil {
+                               if err, ok := err.(*pq.Error); ok &&
+                                       err.Code == "23505" && // unique_violation
+                                       err.Constraint == "ix_pgpkey_fingerprint" {
+                                       valid.Error("We already have this PGP key on file, and duplicates are not
allowed.").
+                                               WithField("pgpKey")
+                                       return errors.New("placeholder")
+                               }
+                               return err
+                       }
+
+                       if _, err := tx.ExecContext(ctx, `
+                                       UPDATE "user" SET pgp_key_id = $1 WHERE id = $2;
+                               `, id, user.ID); err != nil {
+                               return err
+                       }
+               }
+
+               return nil
+       }); err != nil {
+               if !valid.Ok() {
+                       return nil, nil
+               }
+               return nil, err
+       }
+
+       sendRegistrationConfirmation(ctx, &user, pgpKey, confirmation)
+       return &user, nil
+}
+
 func (r *mutationResolver) RegisterOAuthClient(ctx context.Context, redirectURI string, clientName string,
clientDescription *string, clientURL *string) (*model.OAuthClientRegistration, error) {
        var seed [64]byte
        n, err := rand.Read(seed[:])
```

M api/server.go => api/server.go +1 -0

```
@@ 18,6 18,7 @@ func main() {
        appConfig := config.LoadConfig(":5100")

        gqlConfig := api.Config{Resolvers: &graph.Resolver{}}
+       gqlConfig.Directives.Anoninternal = server.AnonInternal
        gqlConfig.Directives.Internal = server.Internal
        gqlConfig.Directives.Private = server.Private
        gqlConfig.Directives.Access = func(ctx context.Context, obj interface{},
```

A metasrht/alembic/versions/8928d88c66d7_add_more_constraints_for_users.py => metasrht/alembic/versions/8928d88c66d7_add_more_constraints_for_users.py +28 -0

```
@@ 0,0 1,28 @@
+"""Add more constraints for users
+
+Revision ID: 8928d88c66d7
+Revises: a08050104214
+Create Date: 2021-09-24 09:13:17.167274
+
+"""
+
+# revision identifiers, used by Alembic.
+revision = '8928d88c66d7'
+down_revision = 'a08050104214'
+
+from alembic import op
+import sqlalchemy as sa
+
+
+def upgrade():
+    op.execute("""
+    ALTER TABLE "user"
+    ADD CONSTRAINT user_email_unique UNIQUE (email);
+    """)
+
+
+def downgrade():
+    op.execute("""
+    ALTER TABLE "user"
+    DROP CONSTRAINT user_email_unique;
+    """)
```

M metasrht/auth/builtin.py => metasrht/auth/builtin.py +2 -2

```
@@ 16,8 16,8 @@ def hash_password(password: str) -> str:


 class BuiltinAuthMethod(AuthMethod):
-    def user_valid(self, valid: Validation, username: str, password: str) \
-            -> bool:
+    def user_valid(self, valid: Validation,
+            username: str, password: str) -> bool:
         username = get_user(username)

         valid.expect(username is not None, "Username or password incorrect")
```

D metasrht/auth_validation.py => metasrht/auth_validation.py +0 -54

```
@@ 1,54 0,0 @@
-import re
-from jinja2 import Markup
-from metasrht.blacklist import email_blacklist, username_blacklist
-from metasrht.types import User
-from srht.config import cfg
-from zxcvbn import zxcvbn
-
-
-def validate_username(valid, username, check_blacklist=True):
-    user = User.query.filter(User.username == username).first()
-    valid.expect(user is None, "This username is already in use.", "username")
-    valid.expect(2 <= len(username) <= 30,
-                 "Username must contain between 2 and 30 characters.",
-                 "username")
-    valid.expect(re.match("^[a-z_]", username),
-                 "Username must start with a lowercase letter or underscore.",
-                 "username")
-    valid.expect(re.match("^[a-z0-9_-]+$", username),
-                 "Username may contain only lowercase letters, numbers, "
-                 "hyphens and underscores", "username")
-    valid.expect(not check_blacklist or username not in username_blacklist,
-                 "This username is not available", "username")
-
-
-def validate_email(valid, email):
-    user = User.query.filter(User.email == email).first()
-    valid.expect(user is None, "This email address is already in use.", "email")
-    valid.expect(len(email) <= 256,
-                 "Email must be no more than 256 characters.", "email")
-    valid.expect("@" in email, "This is not a valid email address.", "email")
-    if valid.ok:
-        [user, domain] = email.split("@")
-        valid.expect(domain not in email_blacklist,
-                     "This email domain is blacklisted. Disposable email "
-                     "addresses are prohibited by the terms of service - we "
-                     "must be able to reach you at your account's primary "
-                     "email address. Contact support if you believe this "
-                     "domain was blacklisted in error.", "email")
-
-
-def validate_password(valid, password):
-    valid.expect(len(password) <= 512,
-                 "Password must be no more than 512 characters.", "password")
-
-    if cfg("sr.ht", "environment", default="production") == "development":
-        return
-    strength = zxcvbn(password)
-    time = strength["crack_times_display"]["offline_slow_hashing_1e4_per_second"]
-    valid.expect(strength["score"] >= 3, Markup(
-        "This password is too weak &mdash; it could be cracked in " +
-        f"{time} if our database were broken into. Try using " +
-        "a few words instead of random letters and symbols. A " +
-        "<a href='https://www.passwordstore.org/'>password manager</a> " +
-        "is strongly recommended."), field="password")
```

M metasrht/blueprints/auth.py => metasrht/blueprints/auth.py +31 -127

```
@@ 6,8 6,6 @@ from metasrht.audit import audit_log
 from metasrht.auth import allow_registration, user_valid, prepare_user
 from metasrht.auth import is_external_auth, set_user_password, set_user_email
 from metasrht.auth.builtin import hash_password, check_password
-from metasrht.auth_validation import validate_password
-from metasrht.auth_validation import validate_username, validate_email
 from metasrht.blueprints.security import metrics as security_metrics
 from metasrht.email import send_email
 from metasrht.totp import totp


@@ 15,6 13,7 @@ from metasrht.types import User, UserType, Invite
 from metasrht.types import UserAuthFactor, FactorType, PGPKey
 from metasrht.webhooks import UserWebhook
 from prometheus_client import Counter
+from srht.crypto import internal_anon
 from srht.config import cfg, get_global_domain
 from srht.database import db
 from srht.flask import csrf_bypass, session


@@ 96,21 95,13 @@ def register():
         return redirect(url_for("auth.register_step2_GET"))
     return render_template("register.html", site_key=site_key_id)

-@auth.route("/register/<invite_hash>")
-def register_invite(invite_hash):
+@auth.route("/register/<invite>")
+def register_invite(invite):
     if current_user:
         return redirect("/")
     if is_external_auth():
         return render_template("register.html")
-
-    invite = (Invite.query
-        .filter(Invite.invite_hash == invite_hash)
-        .filter(Invite.recipient_id == None)
-    ).one_or_none()
-    if not invite:
-        abort(404)
-    return render_template("register.html", site_key=site_key_id,
-            invite_hash=invite_hash)
+    return render_template("register.html", site_key=site_key_id, invite=invite)

 @auth.route("/register", methods=["POST"])
 def register_POST():


@@ 118,162 109,75 @@ def register_POST():

     valid = Validation(request)
     payment = valid.require("payment")
-    invite_hash = valid.optional("invite_hash")
+    invite = valid.optional("invite")
     if not valid.ok:
         abort(400)
     payment = payment == "yes"

-    if not is_open:
-        if not invite_hash:
-            abort(401)
-        else:
-            invite = (Invite.query
-                .filter(Invite.invite_hash == invite_hash)
-                .filter(Invite.recipient_id == None)
-            ).one_or_none()
-            if not invite:
-                abort(401)
-
-    if invite_hash:
-        session["invite_hash"] = invite_hash
```

```
+    if invite:
+        session["invite"] = invite
     session["payment"] = payment

     return redirect(url_for("auth.register_step2_GET"))

 @auth.route("/register/step2")
 def register_step2_GET():
-    invite_hash = session.get("invite_hash")
+    invite = session.get("invite")
     payment = session.get("payment", "no")
     if current_user:
         return redirect("/")
-
-    if invite_hash:
-        invite = (Invite.query
-            .filter(Invite.invite_hash == invite_hash)
-            .filter(Invite.recipient_id == None)
-        ).one_or_none()
-        if not invite:
-            abort(404)
-
     return render_template("register-step2.html",
-            site_key=site_key_id, invite_hash=invite_hash, payment=payment)
+            site_key=site_key_id, invite=invite, payment=payment)

 @auth.route("/register/step2", methods=["POST"])
 def register_step2_POST():
     if current_user:
         abort(400)
     is_open = allow_registration()
-    session.pop("invite_hash", None)
+    session.pop("invite", None)
     payment = session.get("payment", False)

     valid = Validation(request)
     username = valid.require("username", friendly_name="Username")
     email = valid.require("email", friendly_name="Email address")
     password = valid.require("password", friendly_name="Password")
-    invite_hash = valid.optional("invite_hash")
-    pgp_key = valid.optional("pgp-key")
-    invite = None
-
-    valid.expect(not email or "@" in email,
-            "Invalid email address", field="email")
-
-    if email and "@" in email:
-        _, domain = email.split("@")
-        try:
-            answer = resolve(domain, "MX")
-            valid.expect("10minutemail.com" not in answer.response.to_text(),
-                    "This email domain is blacklisted. Disposable email "
-                    "addresses are prohibited by the terms of service - we "
-                    "must be able to reach you at your account's primary "
-                    "email address. Contact support if you believe this "
-                    "domain was blacklisted in error.", "email")
-        except:
-            valid.expect(False, "Invalid email address", field="email")
+    invite = valid.optional("invite", default=None)
+    pgpKey = valid.optional("pgpKey", default=None)
+    if not invite:
+        invite = None
+    if not pgpKey:
+        pgpKey = None

     if not valid.ok:
         return render_template("register-step2.html",
-                is_open=(is_open or invite_hash is not None),
+                is_open=(is_open or invite is not None),
                 site_key=site_key_id, payment=payment, **valid.kwargs), 400

     if is_abuse(valid):
```

```
        return redirect("/registered")

-   if not is_open:
-       if not invite_hash:
-           abort(401)
-       else:
-           invite = (Invite.query
-               .filter(Invite.invite_hash == invite_hash)
-               .filter(Invite.recipient_id == None)
-           ).one_or_none()
-           if not invite:
-               abort(401)
-
-   email = email.strip()
-
-   validate_username(valid, username)
-   validate_email(valid, email)
-   validate_password(valid, password)
-
-   if not valid.ok:
-       return render_template("register-step2.html",
-               is_open=(is_open or invite_hash is not None),
-               site_key=site_key_id, payment=payment, **valid.kwargs), 400
-
    allow_plus_in_email = valid.optional("allow-plus-in-email")
    if "+" in email and allow_plus_in_email != "yes":
        return render_template("register-step2.html",
-               is_open=(is_open or invite_hash is not None),
+               is_open=(is_open or invite is not None),
                site_key=site_key_id, payment=payment, **valid.kwargs), 400

-   user = User(username)
-   user.email = email
-   user.password = hash_password(password)
-   user.invites = cfg("meta.sr.ht::settings", "user-invites", default=0)
-
-   db.session.add(user)
-   db.session.commit()
-   audit_log("account registered", user=user)
-
-   pgp = None
-   if site_key_id and pgp_key:
-       # XXX: We may want to move registration into GraphQL @internal, which
-       # would let us set this as the user's PGP key at the same time and
-       # avoid sending them the audit log notification in plaintext
-       resp = exec_gql("meta.sr.ht", """
-       mutation CreatePGPKey($key: String!) {
-           createPGPKey(key: $key) { id }
+   resp = exec_gql("meta.sr.ht", """
+   mutation RegisterAccount($email: String!, $username: String!,
+           $password: String!, $pgpKey: String, $invite: String) {
+       registerAccount(email: $email, username: $username,
+               password: $password, pgpKey: $pgpKey, invite: $invite) {
+           id
        }
-       """, valid=valid, key=pgp_key, user=user)
-       if not valid.ok:
-           return render_template("register.html",
-               site_key=site_key_id,
-               is_open=(is_open or invite_hash is not None),
-               **valid.kwargs), 400
-       pgp = PGPKey.query.get(resp["createPGPKey"]["id"])
-       assert pgp is not None
-
-   send_email("confirm", user.email,
-           f"Confirm your {site_name} account",
-           headers={
-           "From": f"{cfg('mail', 'smtp-from')}",
-               "To": f"{user.username} <{user.email}>",
-               "Reply-To": f"{cfg('sr.ht', 'owner-name')} <{cfg('sr.ht', 'owner-email')}>",
-           }, user=user, encrypt_key=pgp.key if pgp else None,
-           confirmation=user.confirmation_hash)
```

```
-
-    if invite:
-        invite.recipient_id = user.id
-
-    if pgp:
-        user.pgp_key = pgp
-        db.session.add(pgp)
-        audit_log("changed pgp key",
-                f"Set default PGP key to {pgp.fingerprint_hex}", user=user)
+    }
+    """, valid=valid, user=internal_anon, username=username,
+        email=email, password=password, pgpKey=pgpKey, invite=invite)
+    if not valid.ok:
+        return render_template("register-step2.html",
+                is_open=(is_open or invite is not None),
+                site_key=site_key_id, payment=payment, **valid.kwargs), 400

     metrics.meta_registrations.inc()
-    print(f"New registration: {user.username} ({user.email})")
-    db.session.commit()
     return redirect("/registered")

 @auth.route("/registered")
```

M metasrht/blueprints/profile.py => metasrht/blueprints/profile.py +0 -1

```
@@ 1,6 1,5 @@
 from flask import Blueprint, Response, render_template, request, abort
 from flask import redirect, url_for, session
-from metasrht.blueprints.auth import validate_email
 from metasrht.types import User, UserAuthFactor, FactorType
 from srht.config import cfg
 from srht.database import db
```

M metasrht/templates/privacy.html => metasrht/templates/privacy.html +1 -1

```
@@ 39,7 39,7 @@
              name="pgp-key"
              value="{{key.id}}"
              {%if current_user.pgp_key_id == key.id%}checked{%endif%}
-          /> Encrypt with {{key.email}} {{key.key_id}}
+          /> Encrypt with {{key.email}} {{key.fingerprint_hex}}
         </label>
       </div>
       {% endfor %}
```

M metasrht/templates/register-step2.html => metasrht/templates/register-step2.html +13 -9

```
@@ 17,7 17,7 @@
 <p>Registration is disabled because {{cfg("sr.ht", "site-name")}} authentication
   is managed by a different service. Please contact the system administrator
   for further information.</p>
-{% elif allow_registration() or invite_hash %}
+{% elif allow_registration() or invite %}
 {% if cfg("meta.sr.ht::billing", "enabled") == "yes" %}
 <div class="row">
   <div class="col-md-8 offset-md-2">


@@ 42,8 42,8 @@
   <div class="col-md-6 offset-md-3">
     <form method="POST" action="{{url_for("auth.register_step2_POST")}}">
       {{csrf_token()}}
-      {% if invite_hash %}
-      <input type="hidden" name="invite_hash" value="{{invite_hash}}" />
+      {% if invite %}
+      <input type="hidden" name="invite" value="{{invite}}" />
       <div class="alert alert-info">
         You have received a special invitation to join {{cfg("sr.ht",
         "site-name")}}. Sign up here!


@@ 98,16 98,20 @@
       </div>
       {% if site_key %}
       <div class="form-group">
-        <details>
+        <details
+          {% if valid.cls("pgpKey") == "is-invalid" %}
+          open
+          {% endif %}
+        >
           <summary>PGP public key (optional)</summary>
           <textarea
-            class="form-control {{valid.cls("pgp-key")}}"
-            id="pgp-key"
-            name="pgp-key"
+            class="form-control {{valid.cls("pgpKey")}}"
+            id="pgpKey"
+            name="pgpKey"
             style="font-family: monospace"
             rows="5"
             placeholder="gpg --armor --export-options export-minimal --export fingerprint…"
-          >{{pgp_key or ""}}</textarea>
+          >{{pgpKey or ""}}</textarea>
           <small class="form-text text-muted">
             Emails sent from {{cfg("sr.ht", "site-name")}} are
             signed with our PGP key:<br />


@@ 118,7 122,7 @@
             it now you must be able to decrypt the confirmation email to
             complete registration.
           </small>
-          {{valid.summary("pgp-key")}}
+          {{valid.summary("pgpKey")}}
         </details>
       </div>
       {% endif %}
```

M metasrht/templates/register.html => metasrht/templates/register.html +3 -3

```
@@ 17,7 17,7 @@
 <p>Registration is disabled because {{cfg("sr.ht", "site-name")}} authentication
   is managed by a different service. Please contact the system administrator
   for further information.</p>
-{% elif allow_registration() or invite_hash %}
+{% elif allow_registration() or invite %}
 <form
   class="row"
   action="{{url_for("auth.register_POST")}}"


@@ 25,8 25,8 @@
   style="margin-bottom: 0" {# Look. I know. #}
 >
   {{csrf_token()}}
-  {% if invite_hash %}
-  <input type="hidden" name="invite_hash" value="{{invite_hash}}" />
+  {% if invite %}
+  <input type="hidden" name="invite" value="{{invite}}" />
   {% endif %}
   <div class="col-md-5 offset-md-1 event-list">
     <div class="event">
```

M metasrht/templates/user.html => metasrht/templates/user.html +4 -4

```
@@ 206,18 206,18 @@
     <table class="table">
       <thead>
         <tr>
-          <th>Email</th>
-          <th>Key ID</th>
+          <th>ID</th>
+          <th>Fingerprint</th>
           <th>Authorized</th>
         </tr>
       </thead>
       <tbody>
         {% for key in user.pgp_keys %}
         <tr>
-          <td>{{key.email}}</td>
+          <td>{{key.id}}</td>
           <td>
             <details>
-              <summary>{{key.key_id}}</summary>
+              <summary>{{key.fingerprint_hex}}</summary>
               <pre style="max-width: 600px">{{key.key}}</pre>
             </details>
           </td>
```