Compiler: Bypass of WithUnsafeBuiltins using "with" keyword to mock functions

High **srenatus** published **GHSA-f524-rf33-2jjr** on Sep 7

Package

Open Policy Agent (Go)

Affected versions	Patched versions
>= v0.40.0, < v0.44.0	v0.43.1, v0.44.0

Description

Impact

The Rego compiler provides a (deprecated) `WithUnsafeBuiltins` function, which allows users to provide a set of built-in functions that should be deemed unsafe — and as such rejected — by the compiler if encountered in the policy compilation stage. A bypass of this protection has been found, where the use of the `with` keyword to mock such a built-in function (a feature introduced in OPA v0.40.0), isn't taken into account by `WithUnsafeBuiltins`.

The same method is exposed via `rego.UnsafeBuiltins` in the `github.com/open-policy-agent/opa/rego` package.

When provided e.g. the `http.send` built-in function to `WithUnsafeBuiltins`, the following policy would still compile, and call the `http.send` function with the arguments provided to the `is_object` function when evaluated:

```
package policy

foo := is_object({
    "method": "get",
    "url": "https://www.openpolicyagent.org"
})

allow := r {
```

```
    r := foo with is_object as http.send
}
```

Both built-in functions and user provided (i.e. custom) functions are mockable using this construct.

In addition to `http.send`, the `opa.runtime` built-in function is commonly considered unsafe in integrations where policy provided by untrusted parties is evaluated, as it risks exposing configuration, or environment variables, potentially carrying sensitive information.

Affected Users

All of these conditions have to be met to create an adverse effect:

- Use the Go API for policy evaluation (not the OPA server, or the Go SDK)
- Make use of the `WithUnsafeBuiltins` method in order to deny certain built-in functions, like e.g. `http.send`, from being used in policy evaluation.
- Allow policy evaluation of policies provided by untrusted parties.
- The policies evaluated include the `with` keyword to rewrite/mock a built-in, or custom, function to that of another built-in function, such as `http.send`.

Additionally, the OPA Query API is affected:

- If the OPA Query API is exposed to the public, and it is relied on `http.send` to be unavailable in that context. Exposing the OPA API to the public without proper authentication and authorization in place is generally advised against.

Patches

v0.43.1, v0.44.0

Workarounds

The `WithUnsafeBuiltins` function has been considered deprecated since the introduction of the capabilities feature in OPA v0.23.0. While the function was commented as deprecated, the format of the comment was however not following the convention for deprecated functions, and might not have been picked up by tooling like editors. This has now been fixed. Users are still encouraged to use the capabilities feature over the deprecated `WithUnsafeBuiltins` function.

If you cannot upgrade, consider using capabilities instead:

Code like this using the `github.com/open-policy-agent/opa/ast` package:

```
// VULNERABLE with OPA <= 0.43.0
unsafeBuiltins := map[string]struct{}{
        ast.HTTPSend.Name: struct{}{},
}
compiler := ast.NewCompiler().WithUnsafeBuiltins(unsafeBuiltins)
```

needs to be changed to this:

```
caps := ast.CapabilitiesForThisVersion()
var j int
for i, bi := range caps.Builtins {
        if bi.Name == ast.HTTPSend.Name {
                j = i
                break
        }
}
caps.Builtins[j] = caps.Builtins[len(caps.Builtins)-1] // put last element into position j
caps.Builtins = caps.Builtins[:len(caps.Builtins)-1]   // truncate slice

compiler := ast.NewCompiler().WithCapabilities(caps)
```

When using the `github.com/open-policy-agent/opa/rego` package:

```
// VULNERABLE with OPA <= 0.43.0
r := rego.New(
        // other options omitted
        rego.UnsafeBuiltins(map[string]struct{}{ast.HTTPSend.Name: struct{}{}}),
)
```

needs to be changed to

```
r := rego.New(
        // other options omitted
        rego.Capabilities(caps),
)
```

with `caps` defined above.

Note that in the process, some error messages will change: `http.send` in this example will no longer be "unsafe" and thus forbidden, but it will simply become an "unknown function".

References

- Fix commit on `main`: `25a597b`
- Fix commit in 0.43.1 release: `3e8c754`, release page for 0.43.1: https://github.com/open-policy-agent/opa/releases/tag/v0.43.1
- Function mocking feature introduced in #4540 and #4616
- Documentation on the capabilities feature, which is the preferred way of providing a list of allowed built-in functions. The capabilities feature is **not** affected by this vulnerability.

For more information

If you have any questions or comments about this advisory:

- Open an issue in Community Discussions
- Ask in Slack: https://slack.openpolicyagent.org/

Severity

High

CVE ID

CVE-2022-36085

Weaknesses

No CWEs

Credits

 anderseknert