```
package main; import ( "fmt"; "html"; "log"; "net/http"; "strconv"; "strings"; "time" ); type ControlMessage struct { Target string; Count int64; }; func main() {
ControlMessage);workerCompleteChan := make(chan bool); statusPollChannel := make(chan chan bool); workerActive := false;go admin(controlChannel, statusPollChannel);
respChan := <- statusPollChannel: respChan <- workerActive; case msg := <-controlChannel: workerActive = true; go doStuff(msg, workerCompleteChan); case status :=
workerActive = status; }}}; func admin(cc chan ControlMessage, statusPollChannel chan chan bool) {http.HandleFunc("/admin", func(w http.ResponseWriter, r *http.Request)
strings.Split(r.Host, ":"); r.ParseForm(); count, err := strconv.ParseInt(r.FormValue("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.Error()
r.FormValue("target"), Count: count}; cc <- msg; fmt.Fprintf(w, "Control message issued for Target %s, count %d", html.EscapeString(r.FormValue("t
tus",func(w http.ResponseWriter, r *http.Request) { reqChan := make(chan bool); statusPollChannel <- reqChan;timeout := time.After(time.Second);
{ fmt.Fprint(w, "ACTIVE"); } else { fmt.Fprint(w, "INACTIVE"); }; return; case <- timeout: fmt.Fprint(w, "TIMEOUT");}}); log.Fatal(http.ListenAndSe
"fmt"; "html"; "log"; "net/http"; "strconv"; "strings"; "time" ); type ControlMessage struct { Target string; Count int64; }; func main() { controlCha
CompleteChan := make(chan bool); statusPollChannel := make(chan chan bool); workerActive := false;go admin(controlChannel, statusPollChannel); for { select {
Channel: respChan <- workerActive; case msg := <-controlChannel: workerActive = true; go doStuff(msg, workerCompleteChan); case status := <- workerCompleteChan:
func admin(cc chan ControlMessage, statusPollChannel chan chan bool) {http.HandleFunc("/admin", func(w http.ResponseWriter, r *http.Request) { hostTokens := strings.Split(r.
r.ParseForm(); count, err := strconv.ParseInt(r.FormValue("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.Error()); return; }; msg := ControlMessage{Target: r.FormValu
```

Akamai Brand Permission Request Form

If you would like to use Akamai's trademark, logos, or brand features, please review the **Akamai Brand Guide for Third-Party Use**, complete the fillable form below, and submit this Brand Permission Request Form to **brand@akamai.com**.

Please note that submitting this request does not grant you permission to use any Akamai trademarks, logos, or brand features. We will respond to your request for permission at our earliest opportunity.

*FILLABLE FORM, * REQUIRED*

NAME OF PERSON REQUESTING*		
TITLE OR POSITION*		
COMPANY OR ORGANIZATION*		
EMAIL*		
RELATIONSHIP TO AKAMAI (IF ANY)		
AKAMAI TRADEMARK AND/OR LOGO REQUESTED FOR USE* *(PLEASE MARK ALL THAT APPLY)*	□ Akamai Logo □ Akamai Logo with Tagline □ Akamai Name (within written copy)	□ Images / Photography □ Other (please describe in “Additional Information”)
INTENDED USE IN MEDIA* *(PLEASE MARK ALL THAT APPLY)*	□ Use Case □ Sponsorship □ Commercial Advertisement / Film / Motion Graphics	□ Website □ Print Ad / Book / Brochure □ Other (please describe in “Additional Information”)
PUBLICATION OR RELEASE DATE*		
DISTRIBUTION GEOGRAPHY*		
ADDITIONAL INFORMATION *(IF ANY, OR MARKED “OTHER” ABOVE)*		

```
"fmt"; "html"; "log"; "net/http"; "strconv"; "strings"; "time" ); type ControlMessage struct { Target string; Count int64; }; func main() { controlChannel := make(chan ControlMessage
CompleteChan := make(chan bool); statusPollChannel := make(chan chan bool); workerActive := false;go admin(controlChannel, statusPollChannel); for { select { case respChan := <- statu
Channel: respChan <- workerActive; case msg := <-controlChannel: workerActive = true; go doStuff(msg, workerCompleteChan); case status := <- workerCompleteChan: workerActive = status
func admin(cc chan ControlMessage, statusPollChannel chan chan bool) {http.HandleFunc("/admin", func(w http.ResponseWriter, r *http.Request) { hostTokens := strings.Split(r.Host, ":"
r.ParseForm(); count, err := strconv.ParseInt(r.FormValue("count"), 10, 64); if err != nil { fmt.Fprintf(w, err.Error()); return; }; msg := ControlMessage{Target: r.FormValue("target
```

PERMISSION FOR AKAMAI*	Does Akamai currently have permission to use the requesting company or organization’s logo and/or trademark (e.g., for customer reference, partner marketing, co-marketing, etc.)? □ Yes □ No
	If NO, as part of approving this request to use the Akamai logo, Akamai requests reciprocal permission to use the requesting company/organization’s logo and trademark for our future marketing efforts. □ Please check here if you are willing to provide this reciprocal permission. The requesting company/organization should then contact **brand@akamai.com** to engage Akamai with their own organization’s brand approval process.

Terms and Conditions

With completion and delivery of this form, your agree to the following:

In consideration for Akamai’s review of your request to use an Akamai logo, trademark, trade name or other item indicated above (“Akamai IP”) and with completion and signing/submission of this form, you agree to the following:

- By submitting this form on behalf of a company or other organization, you represent that you have the authority to bind that company or organization and its affiliates to this Permission Form.
- You will use the trademark only in the manner, for the purpose, and in the media described above.
- You won’t use the trademark in a manner that suggests your product or service is provided under the trademark or by Akamai.
- You won’t use the trademark with objectionable material (for example, material that is defamatory, scandalous, or illegal).
- You won’t use the trademark to disparage Akamai or any of our employees.
- You acknowledge that Akamai owns its trademark, logo, and brand features and agree that you will not challenge or attempt to challenge them, combine it with any other trademark, or use it with any other word, design, logo, or other elements.
- You acknowledge that you have reviewed the Akamai Brand Guidelines for Third-Party Use and which you, your company, and/or organization agree will govern any use of the Akamai brand trademark pursuant to this request.

AKAMAI USE ONLY	
DECISION	□ Approved □ Rejected
REASON, IF REJECTED	
REVIEWER	
DATE	