synrion-xbind-readiness-checklist-de

> **Info**
>
> **Letzte freigegebene PDF-Version:**
> synrion-xbind-readiness-checklist-de_v1.0.pdf
> *(Pfad: de/synrion/xid/_media/synrion-xbind-readiness-checklist-de_v1.0.pdf)*

SYNRION x.BIND readiness checkliste

Inhaltsverzeichnis

Änderungsverlauf

- v0.1 – Initialer Entwurf# SYNRION x.BIND

Readiness Checkliste SYNRION x.BIND

Automatisiertes Applikations- und Service-Binding (DORA-ready)

Note

> **Highlight:** SYNRION x.BIND liefert jederzeit fuer jeden Service/Endpoint und jede Applikation den aktuellen Binding-Status (Ist vs. Soll) inklusive Evidence. Zusaetzlich liefert x.BIND proaktives CA Service Monitoring (z.B. ADCS) als Betriebs- und Audit-Signal mit Status, Abweichungen und auditfaehigen Events, bevor daraus Stoerungen oder Audit-Findings werden.

Warum das Thema jetzt kritisch ist (DORA in normaler Sprache)

DORA verlangt, dass kritische IT-Services auch unter Stress stabil laufen und dass Risiken kontrolliert, nachweisbar und auditfaehig behandelt werden.

Im Zertifikatsbetrieb heisst das praktisch:

- Sie muessen ein aktuelles Kryptoregister fuehren: ein Register fuer alle Zertifikate und zertifikatsspeichernden Komponenten (mindestens fuer ICT Assets, die kritische oder wichtige Funktionen unterstuetzen). Das Register muss laufend aktuell gehalten werden. (DORA RTS Art. 7(4))
- Sie muessen Zertifikate rechtzeitig vor Ablauf erneuern und kontrolliert umstellen, damit kritische Services nicht ausfallen. (DORA RTS Art. 7(5))
- Sie muessen nachvollziehbar belegen koennen, was wann wo aktiv gebunden war und welche Aenderungen stattgefunden haben (Evidence fuer Audit/Revision).

Autoenrollment oder "Zertifikate sind irgendwo vorhanden" loest diese Betriebsfrage nicht. Entscheidend ist das Binding am Service/Endpoint, inklusive Trust/Chain und sauberer Evidence.

Zweck

Diese Checkliste klaert in 10-15 Minuten die wichtigsten Fragen vor einer Einfuehrung von SYNRION x.BIND:

- Was kann x.BIND in unserer Umgebung konkret automatisieren (Binding, Renewal, Rollover)?
- Welche Voraussetzungen muessen erfuellt sein (Netz, Rechte, Segmentierung, Prozesse)?
- Welche Sicherheits- und Compliance-Mehrwerte entstehen (Evidence, Audit, DORA-relevante Kontrollen)?
- Wie definieren wir einen sinnvollen Pilotumfang (Systeme, Services, Erfolgskriterien)?

Fokus ist operativer Betrieb: Nicht "Zertifikate ausstellen", sondern Binding, Kontrolle und Nachweis.

Welche Applikation nutzt welches Zertifikat an welchem Service/Endpoint, und ist das Binding jederzeit korrekt, kontrolliert und nachvollziehbar?

Was Sie am Ende haben

- eine kurze Standortbestimmung (Ja/Teilweise/Nein) fuer x.BIND-Faehigkeit und Nutzen
- die Top-Gaps und Risiken, die heute Ausfaelle oder Auditkosten treiben
- eine Liste offener Produkt- und Architekturfragen fuer die Kaufentscheidung
- einen konkreten Pilotvorschlag (20-50 Systeme, 3-5 kritische Services)

Fuer wen lohnt sich SYNRION x.BIND

SYNRION x.BIND lohnt sich besonders, wenn Zertifikatsbetrieb nicht nur "ausstellen", sondern operativ stabil und nachweisbar laufen muss.

ROI und Einstiegsschwelle

x.BIND lohnt sich typischerweise ab ca. 50 Servern oder sobald Zertifikatslaufzeiten von 1 Jahr oder kuerzer geplant sind.
Der Grund ist einfach: Mit kuerzeren Laufzeiten steigen Renewal- und Re-Binding-Zyklen stark an. In manuellen Betriebsmodellen wachsen Aufwand, Fehlerrisiko und Auditkosten ueberproportional.
x.BIND automatisiert Binding, Renewal/Rollover/Re-Binding und liefert Evidence, wodurch der wiederkehrende manuelle Anteil weitgehend entfellt. (Details und Beispielrechnung: Featurehandout SYNRION x.BIND)

Minimalanforderungen (einfaches Setup)

- 1x Windows Server (Windows Server 2019 oder hoeher) fuer x.BIND Core/Service und Portal
- 1x SQL Datenbank fuer Konfiguration, Status, Events und Evidence

Hinweis:

- Minimalsetup kann auf einem Host betrieben werden (je nach Vorgaben auch kombiniert).
- Deployment ist flexibel: klassischer Serverbetrieb, Integration in bestehende Plattformen
- Die Integration in Service-Appliance-Szenarien ist machbar.

Maximalsetup (hoch segmentiert, reguliert, HA)

Fuer hoch segmentierte und regulierte Umgebungen kann x.BIND als HA-Design betrieben werden, inkl. Frontend-Connectoren fuer unterschiedliche Netzwerksegmente.

Typisches Zielbild:

- x.BIND Core/Service und Portal als hochverfuegbare Komponente (HA)
- SQL Datenbank hochverfuegbar (z.B. Cluster/AlwaysOn je nach Plattformstandard)
- Segment-Connectoren (Frontend-Connectoren) pro Zone/Segment, um Binding, Discovery und Checks dort auszufuehren, wo direkte Netzwege bewusst eingeschraenkt sind
- Zentrale Policy/Baseline im Core Backend, dezentrale Ausfuehrung in den Segmenten, zentrale Evidence und Reports

Ergebnis:

- Betrieb auch bei strikter Segmentierung und Minimal-Netzfreigaben
- saubere Trennung von Zonen (Prod, DMZ, Mgmt, Security, etc.)
- kontrollierte Renewals/Rollovers und Evidence ueber alle Segmente hinweg, ohne "Sonderwege" im Netzwerk

---

Struktur der Checkliste (Kurzuebersicht)

1. Quick Pass (10 Fragen): Passt x.BIND zu Ihrer Realitaet?
2. x.BIND Core Use-Cases: Was wird automatisiert (Binding, Renewal, Rollover, Desired State)?
3. Voraussetzungen und Architekturfragen: Was brauchen wir dafuer?
4. Sichtbarkeit, Evidence, Audit: Was bekommen wir heraus?
5. Pilot Worksheet: Scope, Erfolgskriterien, Deliverables
6. Auswertung: Scoring und Interpretation
7. Minimal Evidence Set: was im Audit wirklich zaehlt
8. Pilot Output Paket: was ein Pilot liefern muss

Wie SYNRION x.BIND funktioniert (kurz und einfach)

SYNRION x.BIND automatisiert nicht die Ausstellung von Zertifikaten, sondern deren sicheren Betrieb am Service-Endpunkt.

Im Kern macht x.BIND drei Dinge:

- Es findet Applikationen und Service/Endpoints.
- Es erkennt den Ist-Zustand der aktiven Bindings.
- Es setzt einen kontrollierten Soll-Zustand durch (Desired State).

X.509 Security Baseline (Policy)

Im Core Backend definiert x.BIND eine X.509 Security Baseline.
Diese Baseline kann global, pro Gruppe oder pro Host festgelegt werden.

Der Status wird immer aus Sicht des Systems angezeigt:

- konform

- kritisch
- abweichend
  inklusive klarer Massnahmenempfehlung.

Resilienz und Portal

x.BIND arbeitet auch bei gestoerten Netzwerkverbindungen weiter.
Binding, Kontrolle und Lifecycle laufen lokal; Status und Events werden spaeter mit dem Portal synchronisiert.

Das Portal visualisiert alle Zustaende und liefert verstaendliche Informationen sowie konkrete Loesungsvorschlaege, statt reiner Errordumps.

Protokoll- und Speicherunabhaengig

x.BIND nutzt vorhandene Zertifikate, egal wie sie bereitgestellt werden (z.B. ACME, CEP/CES, Autoenrollment, manuell) und egal wo sie liegen (Windows Store, Filesystem, Java Keystore).

Durch automatisiertes Renewal, Rollover und Re-Binding ermoeglicht x.BIND sehr kurze Laufzeiten und damit hohe Kryptoagilitaet.

Nebenbei, aber entscheidend: CA Service Monitoring (z.B. ADCS)

x.BIND ueberwacht nicht nur Zertifikate und Bindings, sondern auch die beteiligten Services auf dem Host.

Dazu gehoeren auch CA-Komponenten wie ADCS, weil auch diese am Ende Applikationen/Services im Betrieb sind.

Ergebnis:

- Status und Verfuegbarkeit relevanter CA-Services werden sichtbar
- Abweichungen, Fehlerbilder und risikorelevante Ereignisse werden proaktiv erkannt
- auditfaehige Events und Logs koennen zentral nachvollzogen und exportiert werden

Damit wird CA Monitoring vom reaktiven Troubleshooting zu einem proaktiven Betriebs- und Audit-Signal.

Kurz gesagt (Funktionsprinzip)

- Discovery: Applikationen, Service/Endpoints und Zertifikatsspeicher erkennen
- Ist-Zustand: aktives Binding je Service/Endpoint sichtbar machen
- Desired State: Bindings regelbasiert korrekt setzen und stabil halten (z.B. SAP, IIS, RDP, Java Keystore, SQL)
- Lifecycle: Renewal/Rollover/Re-Binding kontrolliert ausfuehren, auch in HA/Cluster
- Trust/Chain: vor Aktivierung pruefen, ob Chain und Trust passen (Trust First, Binding Second)
- Evidence: Events und Historie erzeugen (wer/was/wann), Fingerprints/Hashes, Reports fuer Audit/Revision
- CA Service Monitoring: Status und Abweichungen kritischer CA-Services (z.B. ADCS) proaktiv erkennen und als Evidence bereitstellen

Merksatz:

- Enrollment bringt das Zertifikat.
- x.BIND sorgt dafuer, dass der Service es korrekt nutzt, es rechtzeitig wechselt und das jederzeit nachweisbar ist.

Was ein Zertifikats-Binding wirklich umfasst

Ein Zertifikats-Binding ist mehr als "Zertifikat installieren". Ein Binding umfasst immer den kompletten Betriebskontext:

- Service/Endpoint und Binding-Konfiguration (Zertifikat, Instanz, Port, Listener; ggf. Firewall-Freigaben)
- Private-Key-Zugriff und Schutz (ACLs, optional HSM-backed)
- Zertifikatskette und Trust Anchors (Intermediate/Root) auf System und relevanten Komponenten
- Pruefung von PKI-Abhaengigkeiten vor Aktivierung (AIA, CRL, OCSP)
- kontrolliertes Umschalten bei Renewal/Rollover, auch in HA/Cluster
- Evidence fuer Audit/Revision (Historie, Zeitpunkte, Fingerprints/Hashes, exportierbare Reports)

Offene Fragen, die vor einer Kaufentscheidung geklaert werden sollten

Diese Fragen sind bewusst enthalten, weil sie in der Praxis ueber Erfolg, Sicherheit und Betriebskosten entscheiden.

Architektur und Segmentierung

- Muss x.BIND direkt auf den Zielsystemen laufen (Agent), oder gibt es zentrale Komponenten?
- Welche Netzwege sind notwendig (vom x.BIND Service zu Hosts, Stores, Applikationen)?
- Brauchen wir Segmentierung (z.B. Admin-Netz, Management-Zonen, DMZ), und wie integriert sich x.BIND dort?
- Wie gehen wir mit DMZ-Endpoints um (Reverse Proxy, Load Balancer, Web Gateways)?
- Gibt es HA/Cluster-Anforderungen und wie wird Failover abgedeckt?

PKI, CA, Protokolle

- Welche CAs muessen angebunden werden (ADCS, Public CA, interne SubCAs, mehrere PKIs)?
- Welche Enrollment-Wege sind relevant (ACME, EST, SCEP, Autoenrollment, manuell)?
- Muessen wir an Templates/Policies etwas aendern, oder kann x.BIND bestehende Vorgaben uebernehmen?
- Wie wird ein CA-Wechsel oder Parallelbetrieb mehrerer CAs behandelt (CA Switch)?

Schluessel, HSM und Schutzbedarf

- Brauchen wir HSMs fuer unsere Use-Cases, oder reicht Software Key Storage?
- Welche Systeme muessen zwingend HSM-backed Keys nutzen (kritische Services, Signatur, TLS Termination)?
- Wie wirkt sich das auf Binding, Renewal und Betrieb aus (z.B. Touchpoints, Betriebskonzept)?
- Wie werden Private Keys behandelt (wo entstehen sie, wo liegen sie, wer hat Zugriff)?

Betrieb, Rollen, Rechte, Prozesse

- Welche Rechte braucht x.BIND auf den Zielsystemen (Local Admin, Service Accounts, delegated rights)?
- Wie werden Aenderungen freigegeben (Vier-Augen, Change Management, Maintenance Windows)?

- Welche Rollen gibt es (Owner, Operator, Auditor) und wie wird das technisch enforced?
- Wie wird Drift behandelt: automatische Korrektur vs. nur Alarmierung?

Evidence, Audit, DORA

- Welche Nachweise brauchen Sie im Audit konkret (Binding-Historie, Zeitpunkte, Hashes, Verantwortliche)?
- Wie schnell muessen Reports exportierbar sein (Stunden, Tage)?
- Welche KPIs sind fuer Ihr Management relevant (Coverage, Expiry Risk, Drift, Rollover Success)?
- Wo liegen die groessten Risiken heute (ablaufende Zertifikate, unbekannte Endpoints, manuelle Bindings)?

Kompatibilitaet (Applikationen und Plattformen)

- Welche Applikationen muessen abgedeckt werden (IIS, RDP, SQL, SAP, Java, WinRM, NPS, Reverse Proxies)?
- Welche Plattformen sind im Scope (Windows, Linux, Container, Appliances)?
- Gibt es Sonderfaelle (Legacy, Hardcoded cert paths, proprietaere stores)?

Abschnitt 1: Quick Pass (10 Fragen)

Ziel: In 5 Minuten klaeren, ob SYNRION x.BIND in Ihrer Umgebung sofort Mehrwert liefert und welche Grundvoraussetzungen/offenen Punkte vor einem Pilot zu pruefen sind.

Antworten: Ja / Teilweise / Nein
Notizen: 1 Satz pro Zeile (optional)

#	Frage (kauf- und betriebsrelevant)	Ja	Teilweise	Nein	Notizen
1	Wissen Sie heute fuer Ihre kritischen Services, welches Zertifikat an welchem Service/Endpoint aktiv gebunden ist (nicht nur ausgestellt)?				
2	Gab es in den letzten 12-24 Monaten Stoerungen durch abgelaufene				

#	Frage (kauf- und betriebsrelevant)	Ja	Teilweise	Nein	Notizen
	oder falsch gebundene Zertifikate oder ungeplante Rollover-Events?				
3	Haben Sie mehr als eine PKI/CA (z.B. ADCS plus Public CA, mehrere SubCAs) oder erwarten Sie CA-Wechsel/Parallelbetrieb?				
4	Gibt es HA/Cluster/Load-Balancer-Setups, bei denen Renewals ohne Downtime und ohne manuelle Re-Bindings laufen muessen?				
5	Haben Sie viele manuelle Bindings (IIS/RDP/Java/SQL/SAP/WinRM etc.), die heute als Expertenwissen betrieben werden?				
6	Muessen Sie Evidence im Audit liefern (Binding-Historie, Zeitpunkt,				

#	Frage (kauf- und betriebsrelevant)	Ja	Teilweise	Nein	Notizen
	Hash/Fingerprint, Owner) und ist das heute teuer oder langsam?				
7	Haben Sie Segmentierung/DMZ/Management-Zonen und benoetigen eine Loesung, die dort operativ sauber integrierbar ist?				
8	Gibt es Systeme mit hohem Schutzbedarf, die HSM-backed Keys verlangen (oder in Zukunft verlangen werden)?				
9	Koennen Sie heute einen Desired State fuer Bindings definieren und Abweichungen kontinuierlich erkennen (Policy/Drift)?				
10	Koennen Sie fuer einen Pilot 20-50 Systeme und 3-5 kritische Services benennen (Ow-				

#	Frage (kauf- und betriebsrelevant)	Ja	Teilweise	Nein	Notizen
	ner vorhanden, Zugriff geklaert)?				

Quick Interpretation

- Wenn bei #1, #4, #6 oder #9 mindestens ein "Nein" steht, besteht typischerweise unmittelbarer Nutzen durch x.BIND (Transparenz, Binding-Kontrolle, Evidence, Resilienz).
- Wenn #7 oder #8 "Ja/Teilweise" sind, muessen Architektur- und Security-Fragen frueh geklaert werden (Zonen, Zugriffswege, HSM/Keys).
- Wenn #10 "Nein" ist, fehlt nicht Technik, sondern Scope/Ownership fuer einen sauberen Start.

Abschnitt 2: x.BIND Core Use-Cases (Was wird automatisiert?)

Ziel: Klaeren, welche Binding- und Lifecycle-Use-Cases x.BIND in Ihrer Umgebung abdecken soll und wo heute manueller Aufwand und Risiko entsteht.

Antworten: Ja / Teilweise / Nein
Notizen: kurz und konkret (Applikation/Service nennen)

#	Use-Case / Aussage	Ja	Teilweise	Nein	Notizen
1	TLS/SSL Bindings fuer Web Services (z.B. IIS) automatisch setzen, pruefen und laufend korrekt halten				
2	RDP Zertifikat Binding automatisch setzen und nach Renewal ohne manuellen Eingriff umstellen				
3	Java Keystores (JKS/PKCS12) automatisch ak-				

#	Use-Case / Aussage	Ja	Teilweise	Nein	Notizen
	tualisieren und Applikationen kontrolliert auf neues Zertifikat umstellen				
4	Windows Zertifikatsstores automatisch pflegen und Service Bindings konsistent nachziehen				
5	SQL / AlwaysOn / Listener Zertifikate kontrolliert erneuern und ohne Unterbrechung umstellen				
6	SAP und weitere Enterprise Applikationen mit standardisiertem, nachvollziehbarem Zertifikatswechsel				
7	WinRM und Management Endpoints automatisch versorgen und Bindings stabil halten				
8	Service Discovery: Applikationen, Endpoints und Stores automatisch				

#	Use-Case / Aussage	Ja	Teilweise	Nein	Notizen
	erkennen und inventarisieren				
9	Desired State fuer Bindings definieren und Abweichungen (Drift) automatisch erkennen				
10	Drift Handling nach Policy: automatisch korrigieren oder nur alarmieren und Evidence erzeugen				
11	Renewal Fenster proaktiv steuern und Ablaufrisiken vor Eintritt entfernen				
12	Rollover ohne Downtime: neues Zertifikat vorbereiten, umschalten, verifizieren, dokumentieren				
13	HA/Cluster: Re-Bindings und Rollovers reproduzierbar ohne Service Unterbrechung durchfuehren				

#	Use-Case / Aussage	Ja	Teilweise	Nein	Notizen
14	Multi-CA Betrieb: mehrere interne/externe CAs parallel sauber betreiben, ohne Binding Chaos				
15	CA Switch (Kryptoagilitaet): CA Wechsel als kontrollierter Prozess (Plan, Staging, Cutover, Evidence)				
16	Trust First, Binding Second: fehlende Chain/Trust Anchors verteilen, Schiefstaende korrigieren und erst dann aktiv binden				

Kurzbewertung (Abschnitt 2):

- Wenn #8, #9, #11 oder #12 "Nein" sind, ist der Handlungsdruck hoch: Discovery, Desired State und kontrollierte Renewals/Rollovers fehlen als Basis fuer stabilen Betrieb.
- Wenn #15 oder #16 "Ja/Teilweise" ist, sollten CA Switch und Trust/Chain Rollout explizit in den Pilot, weil hier die meisten Renewal-Risiken und Audit-Findings entstehen.
- Wenn #13 "Ja/Teilweise" ist, muss HA/Cluster frueh als Pilotkriterium gesetzt werden, da Binding und Rollover dort ohne Downtime funktionieren muessen.
- Wenn viele der Punkte #1 bis #7 "Teilweise/Nein" sind, ist der Nutzen typischerweise sofort messbar, weil manueller Binding-Aufwand und Incident-Risiko direkt sinken.

Abschnitt 3: Voraussetzungen und Architekturfragen (Was brauchen wir dafuer?)

Ziel: Klare technische und organisatorische Voraussetzungen fuer einen sicheren Betrieb von x.BIND (Netz, Rechte, Zonen, Prozesse).

Antworten: Ja / Teilweise / Nein
Notizen: konkrete Zonen, Accounts, Prozesse

#	Voraussetzung / Frage	Ja	Teilweise	Nein	Notizen
1	Gibt es ein definiertes Zielbild fuer Segmentierung/Zonen (Admin, Mgmt, Prod, DMZ)?				
2	Sind benoetigte Netzwege zwischen x.BIND Komponenten und Zielsystemen moeglich (Ports, Routing, Firewalls)?				
3	Gibt es eine klare Strategie fuer DMZ Endpoints (z.B. Reverse Proxy, Load Balancer, Gateways)?				
4	Sind Service Accounts/Identitaeten fuer x.BIND definiert (Least Privilege, Delegation)?				
5	Sind benoetigte Rechte auf Zielsystemen klaerbar (Stores, Dienste,				

#	Voraussetzung / Frage	Ja	Teilweise	Nein	Notizen
	Konfig Files, Keystores)?				
6	Change Prozess: duerfen Bindings automatisch umgestellt werden (Freigaben, Maintenance Windows)?				
7	Rollenmodell: Owner/Operator/Auditor klar (wer darf was, wer sieht was)?				
8	Logging/Monitoring: gibt es zentrale Log Ablage und Alarming (SIEM, Ticketing, Mail, Teams)?				
9	Backup/Recovery: sind Zertifikatsstores/Keystores und Konfigs gesichert (Restore faehig)?				
10	HSM Frage: gibt es Systeme mit Pflicht zu HSM-backed Keys oder geplantem HSM Einsatz?				

Kurzbewertung (Abschnitt 3):

- Wenn #2, #4 oder #5 "Nein" ist, muss zuerst Betriebsfaehigkeit (Netz/Rollen/Rechte) geklaert werden.
- Wenn #6 "Nein" ist, ist x.BIND trotzdem nutzbar, aber eher als "Detect + Prove" statt "Auto Fix".

Abschnitt 4: Sichtbarkeit, Evidence, Audit (Was bekommen wir heraus?)

Ziel: Klaeren, ob x.BIND die benoetigten Nachweise und Steuerungsdaten liefert und wie schnell diese bereitstehen muessen.

Antworten: Ja / Teilweise / Nein
Notizen: Audit Anforderungen, Report Formate, SLA

#	Evidence / Frage	Ja	Teilweise	Nein	Notizen
1	Gibt es heute eine zentrale Sicht auf aktive Bindings je Service/Endpoint?				
2	Koennen Sie Binding Historie fuer Service/Endpoint X zu Datum Y belegen (wer/was/wann)?				
3	Gibt es KPIs fuer Coverage (wie viel ist automatisiert/unter Kontrolle)?				
4	Koennen Reports exportiert werden (CSV/Report, bei Bedarf PDF) fuer Audit/Revision innerhalb				

#	Evidence / Frage	Ja	Teilweise	Nein	Notizen
	von Stunden?				
5	Werden PKI Kritikalitaeten sichtbar (Expiry, CRL, OCSP, AIA, Policy Abweichungen)?				
6	Gibt es definierte Eskalationswege bei Risiken (Ticket, On-Call, Mail, Teams)?				
7	Ist Evidence nachvollziehbar und auditfaehig (Change Events, Hash/Fingerprint, Zeitstempel)?				
8	Gibt es eine klare Owner Zuordnung pro kritischem Service (Audit Frage #1)?				

Kurzbewertung (Abschnitt 4):

- Wenn #2 oder #4 "Nein" ist, ist Audit Readiness typischerweise teuer und langsam; x.BIND adressiert genau diese Luecke.
- Wenn #5 "Teilweise/Nein" ist, entstehen Ausfaelle oft nicht durch Ablauf allein, sondern durch Validierungsprobleme (CRL/OCSP/AIA).

Abschnitt 5: Pilot Worksheet (Scope, Erfolgskriterien, Deliverables)

Ziel: In 10 Minuten einen Pilot definieren, der belastbare Ergebnisse liefert.

5.1 Pilot Scope

Ziel:

- 20-50 Systeme
- 3-5 kritische Services (inkl. 1 HA/Cluster Use-Case, falls vorhanden)

Feld	Eintrag
Pilot Owner (Business)	
Pilot Owner (Technical)	
Systeme (Anzahl + Beispiele)	
Kritische Services (3-5)	
Zonen (Prod, DMZ, etc.)	
PKI/CAs im Scope	
Applikationen im Scope (IIS/RDP/Java/SQL/SAP/WinRM...)	

5.2 Erfolgskriterien (Messbar)

KPI / Kriterium	Zielwert	Messmethode
Binding Mapping Coverage		Dashboard/Report
Evidence Export Time		Zeitmessung
Renewal + Re-Bind Success Rate		Events/Report
Drift Findings pro Woche		Drift Report
Zeit bis Drift behoben		Ticket/Workflow

5.3 Deliverables (Was liefern wir im Pilot?)

- Binding Inventory und Mapping (Service/Endpoint -> aktives Zertifikat)
- Risiko Liste (Expiry, Drift, Policy Abweichungen)
- Evidence Paket (Binding Historie + Exporte)
- Empfehlung fuer Rollout (Scope, Rollen, Zonen, Betrieb)

Abschnitt 6: Auswertung (Scoring und Interpretation)

Ziel: Aus Ja/Teilweise/Nein eine klare Aussage ableiten, ohne lange Diskussion.

Scoring:

- Ja = 2
- Teilweise = 1
- Nein = 0

Interpretation:

- 0-25 Punkte: Hoher Handlungsdruck (Binding/Inventory/Evidence sind nicht unter Kontrolle)
- 26-45 Punkte: Mittel (Teilautomation, aber Luecken bei Drift, Renewal, Evidence)
- 46-60 Punkte: Gut (nahe DORA-ready, Fokus auf Skalierung und HA/CA Switch)
- 61+ Punkte: Sehr gut (stabiler Betrieb, klare Evidence, kurze Laufzeiten realistisch)

Gate Fragen (Quick Pass):

- #1, #4, #6, #9 sind Gate Fragen. Ein "Nein" dort ist fast immer ein Pilot-Kandidat.

Score Tabelle (optional):

Bereich	Punkte	Kommentar
Abschnitt 1: Quick Pass (10 Fragen)		
Abschnitt 2: Core Use-Cases (16 Aussagen)		
Abschnitt 3: Voraussetzungen (10 Fragen)		
Abschnitt 4: Evidence (8 Fragen)		
Gesamt		

Abschnitt 7: Minimal Evidence Set (was im Audit wirklich zaehlt)

Ziel: Klar machen, was "auditfaehig" in der Praxis bedeutet (ohne Papier und Screenshots).

Minimal Evidence (sollte in Stunden lieferbar sein):

- Binding Mapping: Service/Endpoint -> aktives Zertifikat (Fingerprint/Hash)
- Binding History: wer/was/wann (Change Event, Zeitpunkt, Endpoint, neues Zertifikat)
- Drift Report: Abweichung zur Baseline (was ist falsch, warum kritisch, Massnahme)
- Expiry Risk: Renewals innerhalb definierter Fenster (z.B. 30/14/7 Tage)
- Trust/Chain Check: Ergebnis vor Aktivierung (OK/Fail, Grund, Massnahme)
- CA Service Monitoring (z.B. ADCS): Status/Abweichungen/Audit Events als Report

Optional (wenn benoetigt):

- OCSP/CRL/AIA Reachability: Abhaengigkeiten sichtbar, Timeouts/Failures erkennbar
- Export: CSV/Report (bei Bedarf PDF)

Abschnitt 8: Pilot Output Paket (was ein Pilot liefern muss)

Ziel: Der Pilot muss messbar sein und ein Ergebnis liefern, das man intern weitergeben kann.

Pilot Outputs (Minimum):

1. Inventory Snapshot

- Liste der Pilot-Systeme
- erkannte Applikationen/Endpoints
- aktives Binding je Service/Endpoint (inkl. Fingerprint/Hash)

2. Risiko und Abweichungen

- Expiry Risiken (Top 10)
- Drift/Policy Abweichungen (Top 10)
- Trust/Chain Findings (Top 10)
- CA Service Findings (z.B. ADCS Status/Abweichungen)

3. Evidence Paket

- Binding History Export (Events)
- Change Nachweise fuer mindestens 1 Renewal/Rollover (vorher/nachher, inkl. Trust/Chain Check)

4. Betriebsempfehlung

- Rollen und Rechte (Minimalmodell)
- Zonen/Segmentierung (Netzwege)
- Rollout Plan (naechste 100/500/5000 Systeme)

Erfolg = wenn 3 Dinge nachweislich besser sind:

- Evidence Export Time sinkt (z.B. Tage -> Stunden)
- Renewal/Re-Bind laeuft kontrolliert (mindestens 1-2 erfolgreiche Rollovers)
- Drift wird sichtbar und beherrschbar (Findings + Massnahmen)